Generic Forward & Backward Simulation III:

Quantitative Simulation by Matrices

Natsuki Urabe, Ichiro Hasuo
The University of Tokyo

Motivation

- Formal verification of **quantitative systems**
  - Verify that given quantitative system satisfies quantitative property
  - e.g. probability, time, energy, etc…

Motivation

- Formal verification of **quantitative systems**
  - Verify that given quantitative system satisfies quantitative property
  - e.g. probability, time, energy, etc…

➡ **Simulation-based verification**

Preliminaries: Simulation-based Verification for Non-deterministic Systems

Preliminaries: Simulation-based Verification for Non-deterministic Systems

Implementation

$\mathcal{I}$

Safety property

$\mathcal{P}$

Preliminaries: Simulation-based Verification for Non-deterministic Systems

non-det.
automaton

Implementation

$\mathcal{I}$

Qualitative

Safety property

$\mathcal{P}$

Preliminaries: Simulation-based Verification for Non-deterministic Systems

non-det.
automaton

Implementation

$\mathcal{I}$

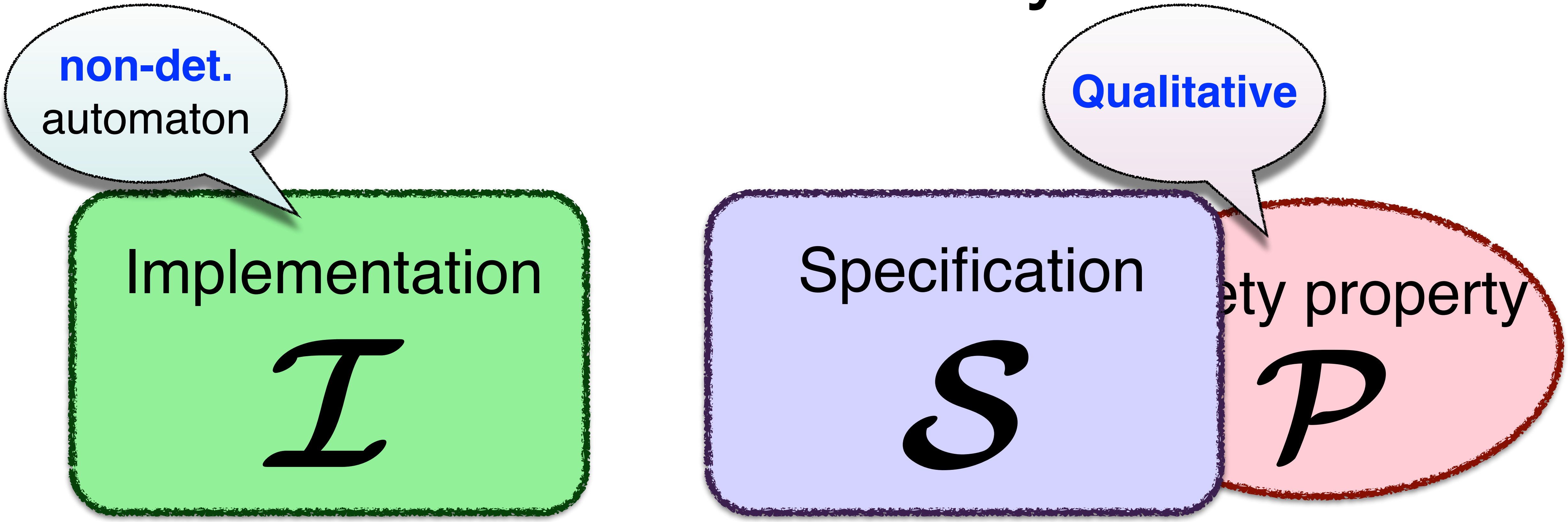
Qualitative

Safety property

$\mathcal{P}$

Aim : Prove $\mathcal{I} \models \mathcal{P}$

Preliminaries: Simulation-based Verification for Non-deterministic Systems



non-det. automaton

Implementation

$\mathcal{I}$

Specification

$\mathcal{S}$

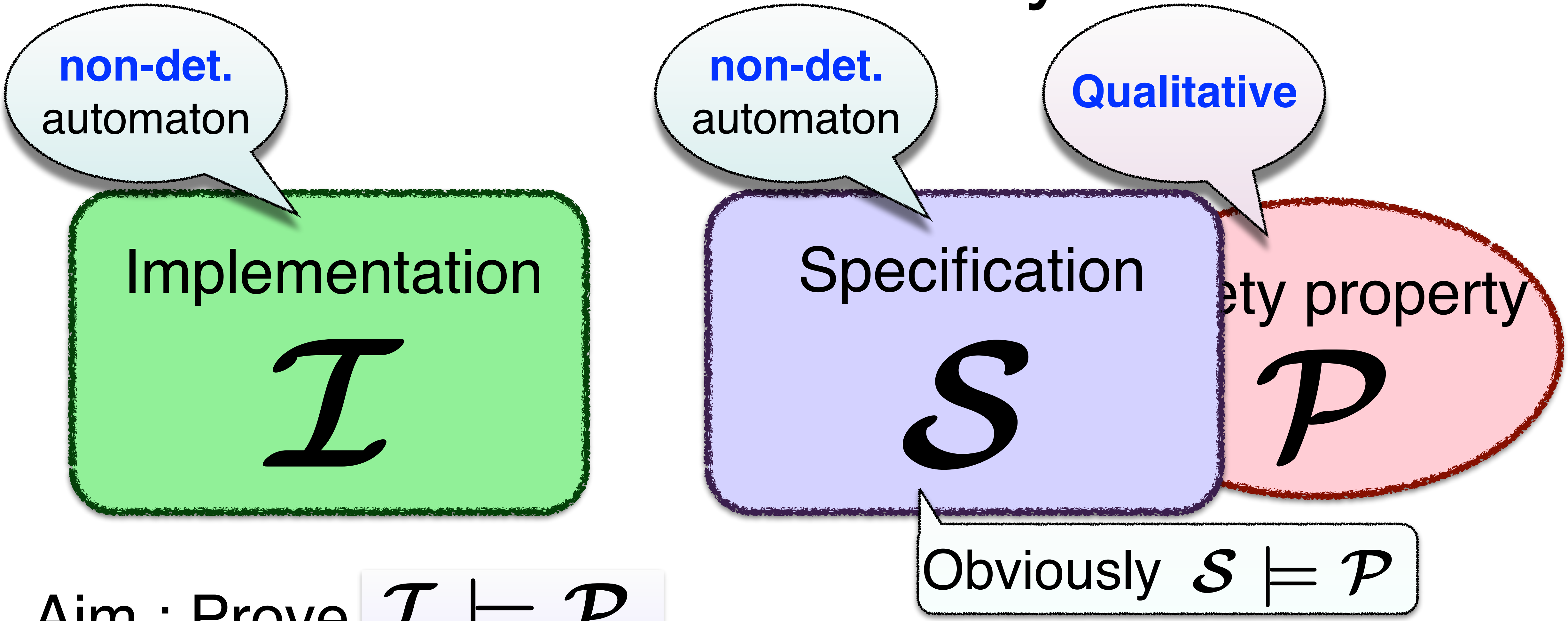
Qualitative

ety property

$\mathcal{P}$

Aim : Prove $\mathcal{I} \models \mathcal{P}$

Preliminaries: Simulation-based Verification for Non-deterministic Systems



non-det. automaton

Implementation

$\mathcal{I}$

non-det. automaton

Specification

$\mathcal{S}$

Qualitative

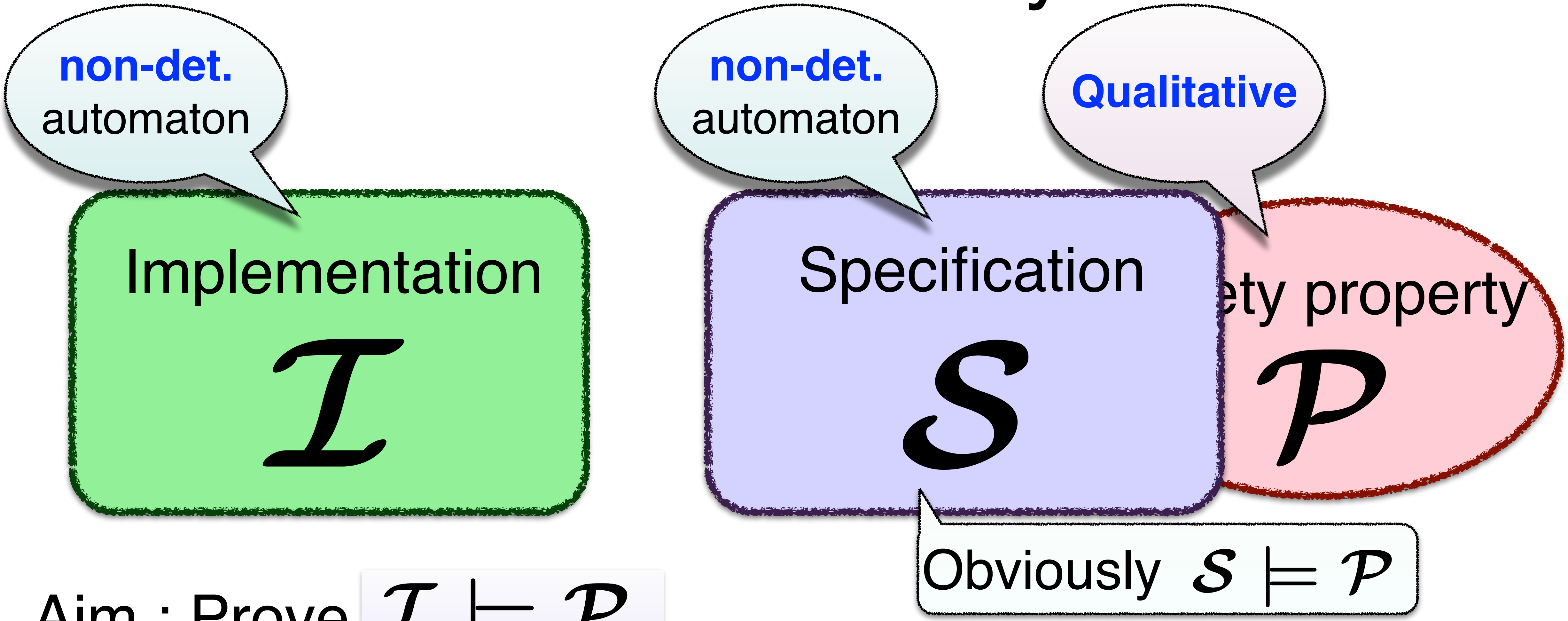
ety property

$\mathcal{P}$

Obviously $\mathcal{S} \models \mathcal{P}$

Aim : Prove $\mathcal{I} \models \mathcal{P}$

Preliminaries: Simulation-based Verification for Non-deterministic Systems



non-det. automaton

Implementation

$\mathcal{I}$

non-det. automaton

Specification

$\mathcal{S}$

Qualitative

ety property

$\mathcal{P}$

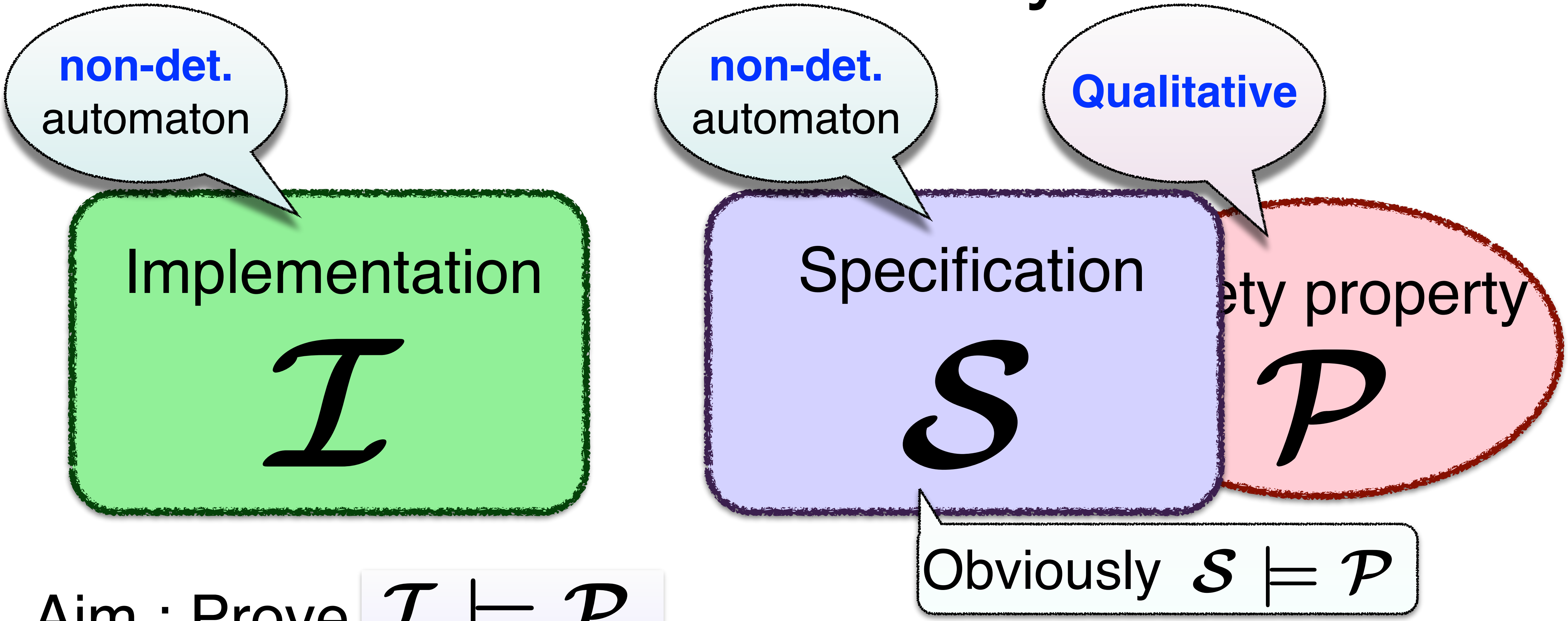
Obviously $\mathcal{S} \models \mathcal{P}$

Aim : Prove $\mathcal{I} \models \mathcal{P}$

Prove $\underline{\mathrm{Lang}(\mathcal{I})}$

Set of possible outputs of non-det. automaton

Preliminaries: Simulation-based Verification for Non-deterministic Systems

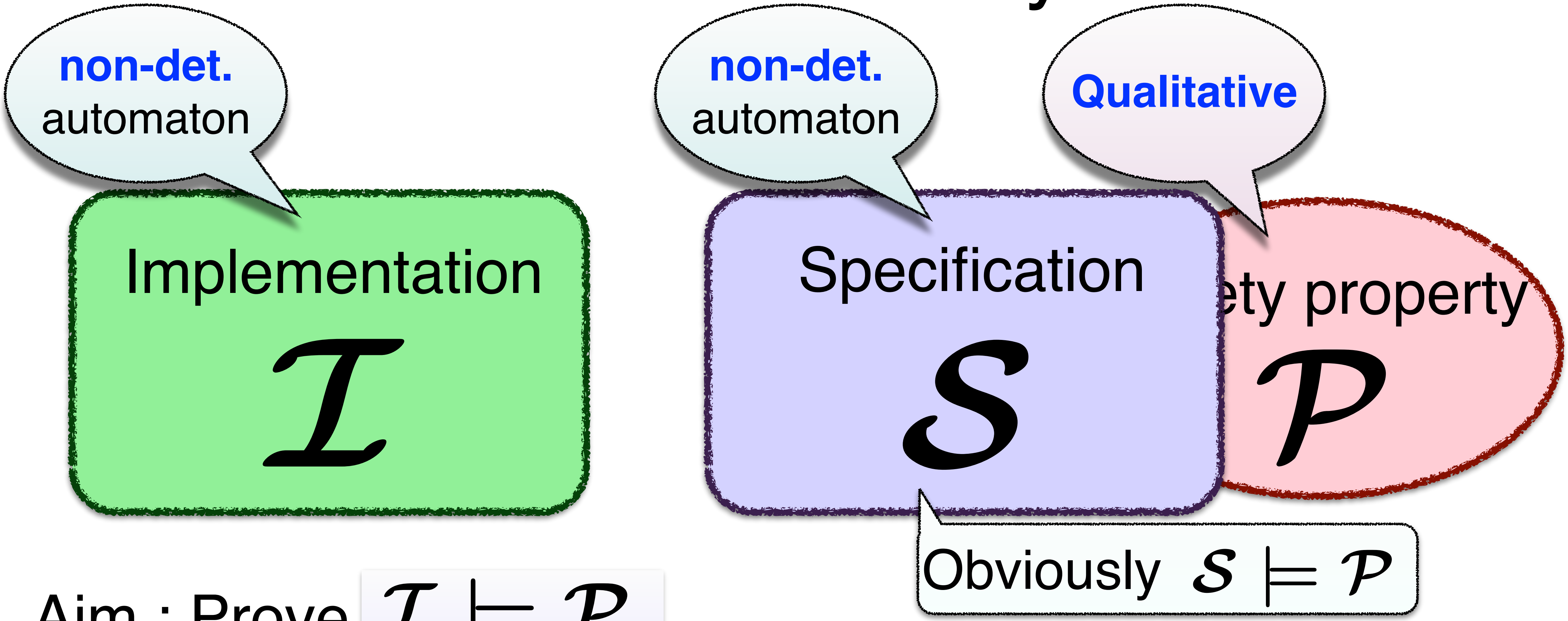


Aim : Prove $\mathcal{I} \models \mathcal{P}$

Prove $\mathrm{Lang}(\mathcal{I}) \subseteq \mathrm{Lang}(\mathcal{S})$ (language inclusion)

Set of possible outputs of non-det. automaton

Preliminaries: Simulation-based Verification for Non-deterministic Systems



non-det. automaton

Implementation

$\mathcal{I}$

non-det. automaton

Specification

$\mathcal{S}$

Qualitative

ety property

$\mathcal{P}$

Obviously $\mathcal{S} \models \mathcal{P}$

Aim : Prove $\mathcal{I} \models \mathcal{P}$

$$\Uparrow$$

Prove $\underline{\mathrm{Lang}(\mathcal{I})} \subseteq \underline{\mathrm{Lang}(\mathcal{S})}$ (language inclusion)

Set of possible outputs of non-det. automaton

Problem

Implementation

$\mathcal{I}$

Specification

$\mathcal{S}$

ety property

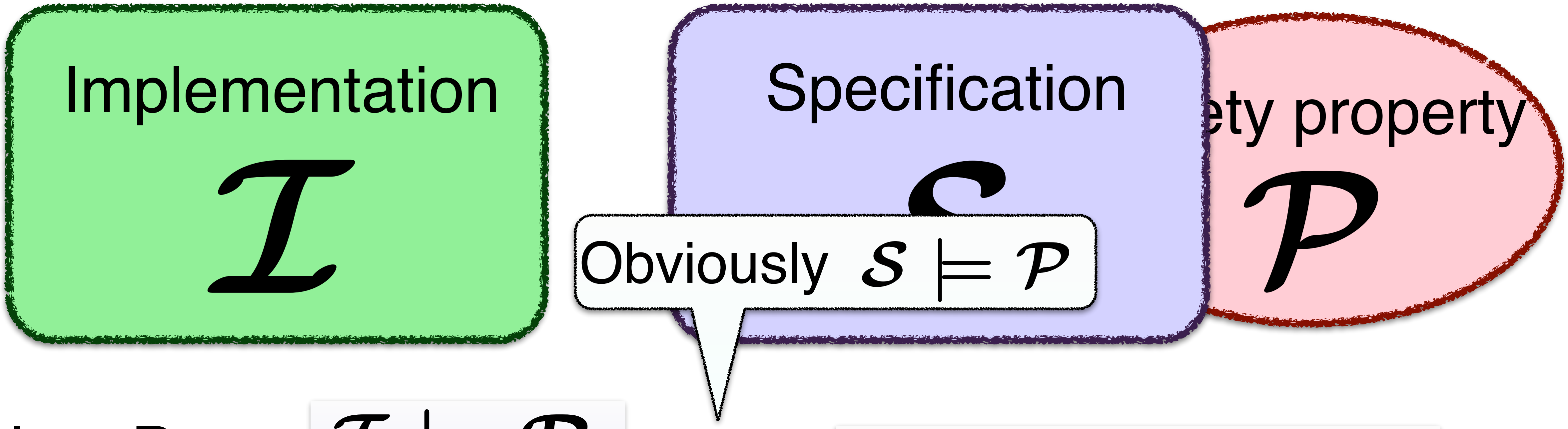
$\mathcal{P}$

Aim : Prove $\mathcal{I} \models \mathcal{P}$

$$\Uparrow$$

$$\mathrm{Lang}(\mathcal{I}) \subseteq \mathrm{Lang}(\mathcal{S})$$

Problem



Implementation

$\mathcal{I}$

Specification

$\mathcal{S}$

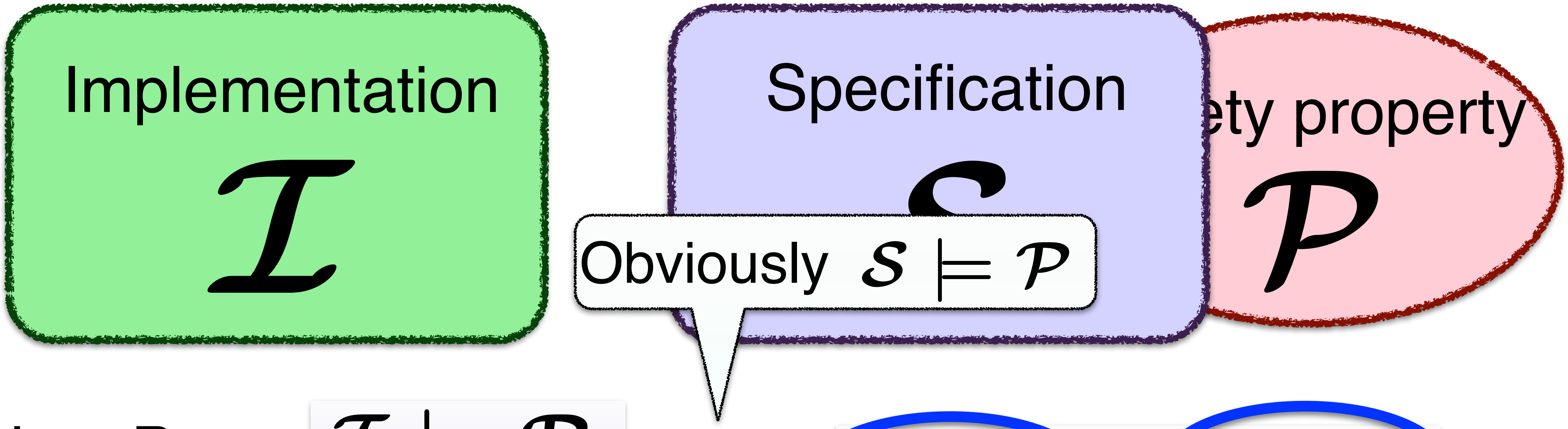
ety property

$\mathcal{P}$

Obviously $\mathcal{S} \models \mathcal{P}$

Aim : Prove $\mathcal{I} \models \mathcal{P}$ $\Leftarrow$ $\mathrm{Lang}(\mathcal{I}) \subseteq \mathrm{Lang}(\mathcal{S})$

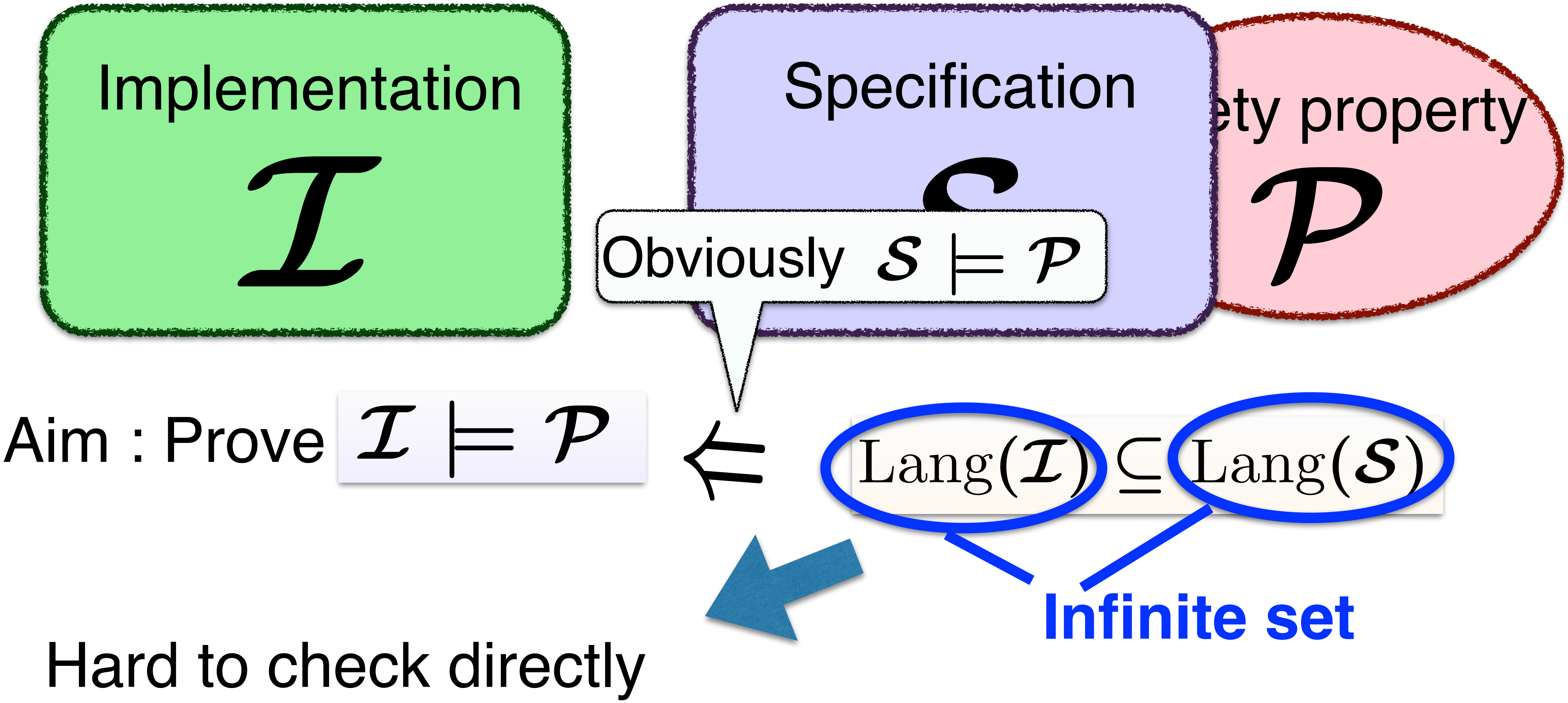
Problem



Aim : Prove $\mathcal{I} \models \mathcal{P}$ $\Leftarrow$ $\mathrm{Lang}(\mathcal{I}) \subseteq \mathrm{Lang}(\mathcal{S})$

Infinite set

Problem



Implementation

$\mathcal{I}$

Specification

$\mathcal{S}$

ety property

$\mathcal{P}$

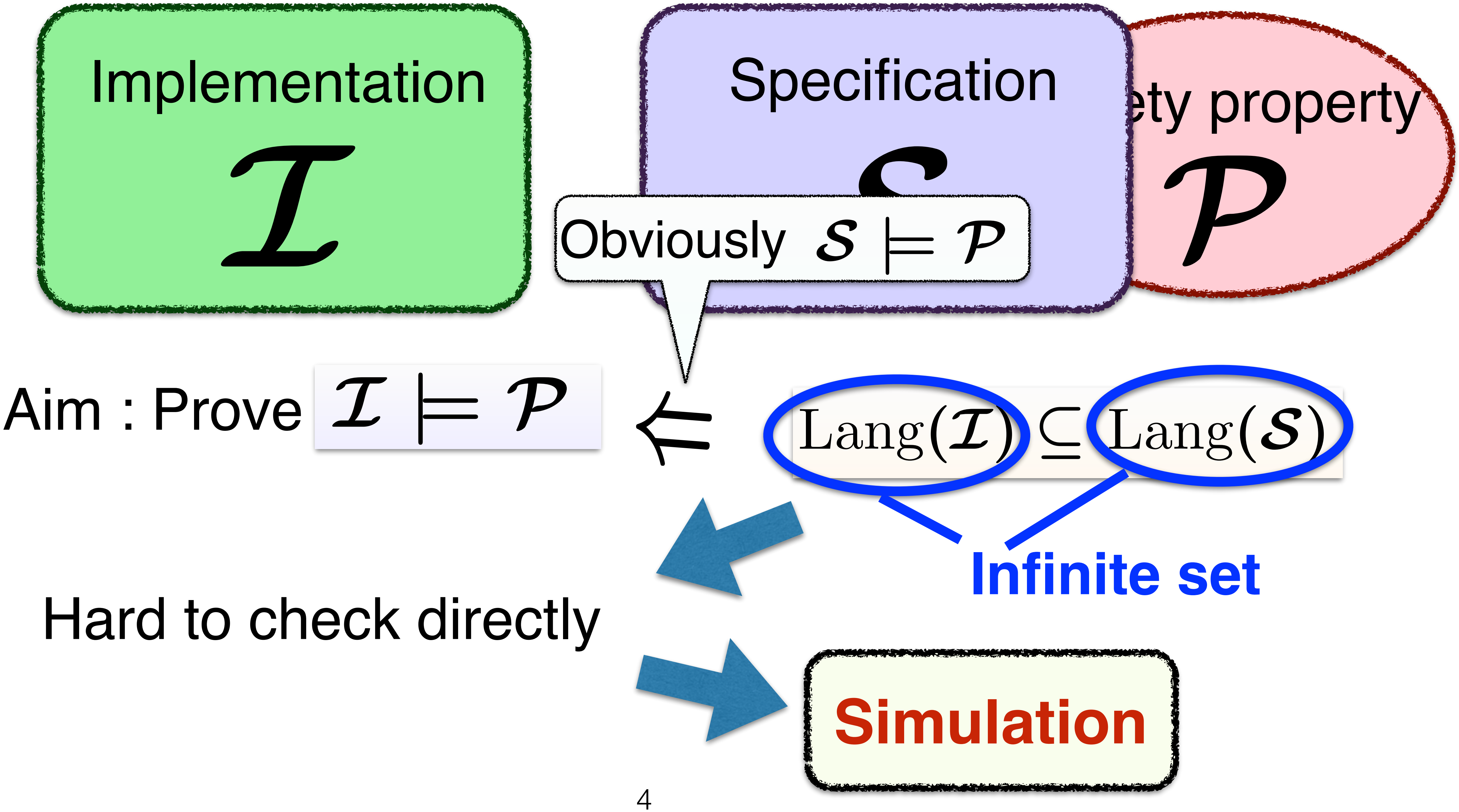
Obviously $\mathcal{S} \models \mathcal{P}$

Aim : Prove $\mathcal{I} \models \mathcal{P}$ $\Lleftarrow$ $\mathrm{Lang}(\mathcal{I}) \subseteq \mathrm{Lang}(\mathcal{S})$

Infinite set

Hard to check directly

Problem



Implementation

$\mathcal{I}$

Specification

$\mathcal{S}$

ety property

$\mathcal{P}$

Obviously $\mathcal{S} \models \mathcal{P}$

Aim : Prove $\mathcal{I} \models \mathcal{P}$ $\Leftarrow$ $\mathrm{Lang}(\mathcal{I}) \subseteq \mathrm{Lang}(\mathcal{S})$

Infinite set

Hard to check directly

Simulation

Simulation-Based Verification

Implementation

$\mathcal{I}$

Specification

$\mathcal{S}$

ety property

$\mathcal{P}$

- Find **simulation** between $\mathcal{I}$ and $\mathcal{S}$ instead

step-wise language inclusion

Simulation-Based Verification

Implementation

$\mathcal{I}$

Specification

$\mathcal{S}$

ety property

$\mathcal{P}$

- Find **simulation** between $\mathcal{I}$ and $\mathcal{S}$ instead

step-wise language inclusion

- Soundness: $\mathcal{I} \sqsubseteq_{\mathrm{sim}} \mathcal{S} \quad \Rightarrow \mathrm{Lang}(\mathcal{I}) \subseteq \mathrm{Lang}(\mathcal{S})$

Example: Fwd. & Bwd. Simulation for Non-deterministic System [Lynch & Vaandrager 1994]

- For two non-det. automata, fwd./ bwd. simulation is relation R between state spaces such that

Example: Fwd. & Bwd. Simulation for Non-deterministic System [Lynch & Vaandrager 1994]

- For two non-det. automata, fwd./ bwd. simulation is relation R between state spaces such that

fwd:

$$\begin{tikzcd} & x' & & \\ \mathcal{A} & x \arrow[u, "a"] \arrow[r, dash, dashed, "R"] & y & \mathcal{B} \end{tikzcd}$$

Example: Fwd. & Bwd. Simulation for Non-deterministic System [Lynch & Vaandrager 1994]

- For two non-det. automata, fwd./ bwd. simulation is relation R between state spaces such that

fwd:

$$\begin{tikzcd} & x' & & \\ \mathcal{A} & x \arrow[u, "a"] \arrow[r, dash, dashed, "R"] & y & \mathcal{B} \end{tikzcd} \Rightarrow \begin{tikzcd} & x' \arrow[r, dash, dashed, "R"] & \exists y' & \\ \mathcal{A} & x \arrow[u, "a"] \arrow[r, dash, dashed, "R"] & y \arrow[u, "a"] & \mathcal{B} \end{tikzcd}$$

Example: Fwd. & Bwd. Simulation for Non-deterministic System [Lynch & Vaandrager 1994]

- For two non-det. automata, fwd./ bwd. simulation is relation R between state spaces such that

fwd:

$$\begin{tikzcd} x' & & & & x' \arrow[r, dash, dashed, "R"] & \exists y' \\ x \arrow[u, "a"] \arrow[r, dash, dashed, "R"] & y & \Rightarrow & & x \arrow[u, "a"] \arrow[r, dash, dashed, "R"] & y \arrow[u, "a"'] \\ \mathcal{A} & \mathcal{B} & & & \mathcal{A} & \mathcal{B} \end{tikzcd}$$

bwd:

$$\begin{tikzcd} x' \arrow[r, dash, "R"] & y' \\ x \arrow[u, "a"] & \\ \mathcal{A} & \mathcal{B} \end{tikzcd}$$

Example: Fwd. & Bwd. Simulation for Non-deterministic System [Lynch & Vaandrager 1994]

- For two non-det. automata, fwd./ bwd. simulation is relation R between state spaces such that

fwd:

$$\begin{tikzcd}
 & x' & & & & & x' \arrow[r, dash, dashed, red, "R"] & \color{red}\exists y' & \\
\mathcal{A} & x \arrow[u, "a"] \arrow[r, dash, dashed, "R"] & y & \mathcal{B} & \Longrightarrow & \mathcal{A} & x \arrow[u, "a"] \arrow[r, dash, dashed, "R"] & y \arrow[u, red, "\color{red}a"'] & \mathcal{B}
\end{tikzcd}$$

bwd:

$$\begin{tikzcd}
 & x' \arrow[r, dash, "R"] & y' & & & & x' \arrow[r, dash, "R"] & y' & \\
\mathcal{A} & x \arrow[u, "a"] & & \mathcal{B} & \Longrightarrow & \mathcal{A} & x \arrow[u, "a"] \arrow[r, dash, dashed, red, "R"] & \color{red}\exists y \arrow[u, red, "\color{red}a"] & \mathcal{B}
\end{tikzcd}$$

Example: Fwd. & Bwd. Simulation for Non-deterministic System [Lynch & Vaandrager 1994]

- For two non-det. automata, fwd./ bwd. simulation is relation R between state spaces such that

$$\begin{tikzcd}
& x' \arrow[r, dash, dashed, "R"] & \exists y' & \\
\mathcal{A} & x \arrow[u, "a"] \arrow[r, dash, dashed, "R"] & y \arrow[u, "a"'] & \mathcal{B}
\end{tikzcd}$$

$$\begin{tikzcd}
& x' \arrow[r, dash, "R"] & y' & \\
\mathcal{A} & x \arrow[u, "a"] \arrow[r, dash, dashed, "R"] & \exists y \arrow[u, "a"'] & \mathcal{B}
\end{tikzcd}$$

Example: Fwd. & Bwd. Simulation for Non-deterministic System [Lynch & Vaandrager 1994]

- For two non-det. automata, fwd./ bwd. simulation is relation R between state spaces such that

fwd:
$$\begin{tikzcd}
 & x' \arrow[r, dash, dashed, "R"] & \exists y' & \\
 & x \arrow[u, "a"] \arrow[r, dash, dashed, "R"] & y \arrow[u, "a"'] & \\
\mathcal{A} & & & \mathcal{B}
\end{tikzcd}$$

bwd:
$$\begin{tikzcd}
 & x' \arrow[r, dash, "R"] & y' & \\
 & x \arrow[u, "a"] \arrow[r, dash, dashed, "R"] & \exists y \arrow[u, "a"'] & \\
\mathcal{A} & & & \mathcal{B}
\end{tikzcd}$$

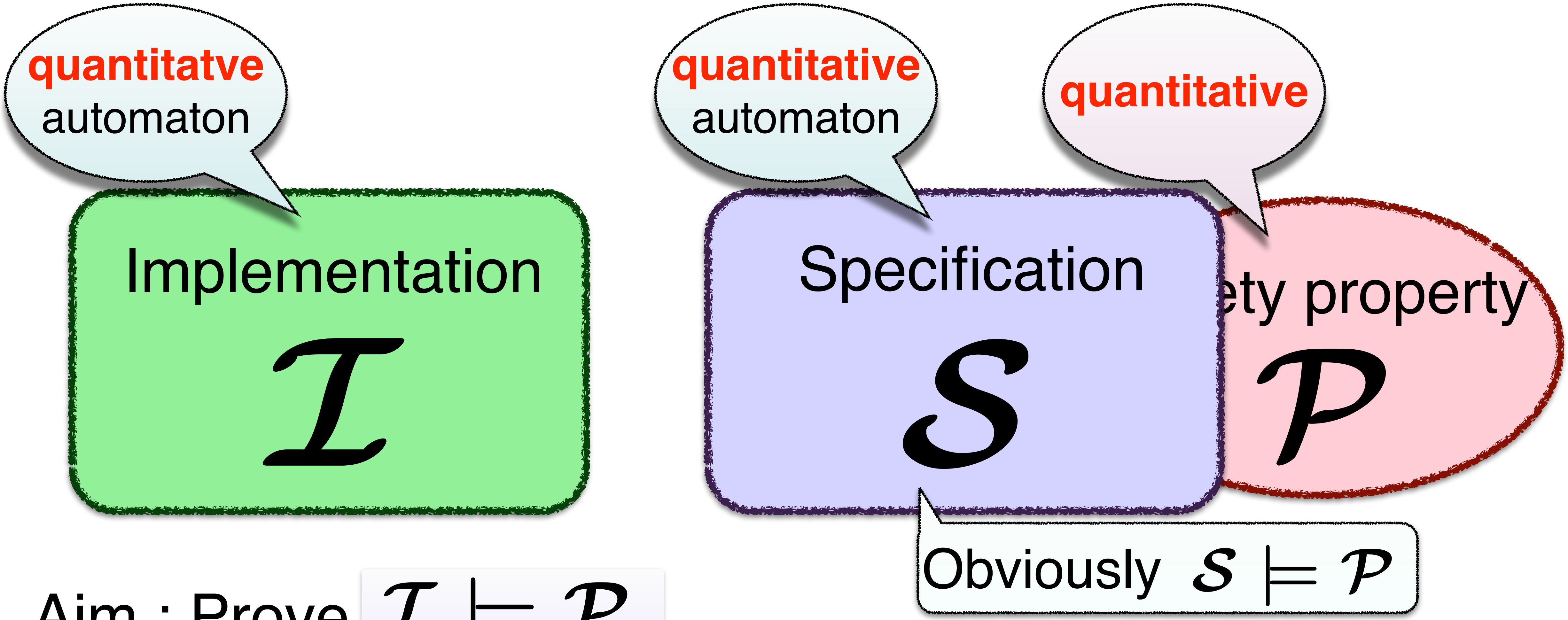
Thm (Soundness) :

A fwd. simulation exists

A bwd. simulation exists

$\Longrightarrow \mathrm{Lang}(\mathcal{I}) \subseteq \mathrm{Lang}(\mathcal{S})$

Simulation-based Verification for Quantitative Systems



quantitatve automaton

Implementation

$\mathcal{I}$

quantitative automaton

Specification

$\mathcal{S}$

quantitative

ety property

$\mathcal{P}$

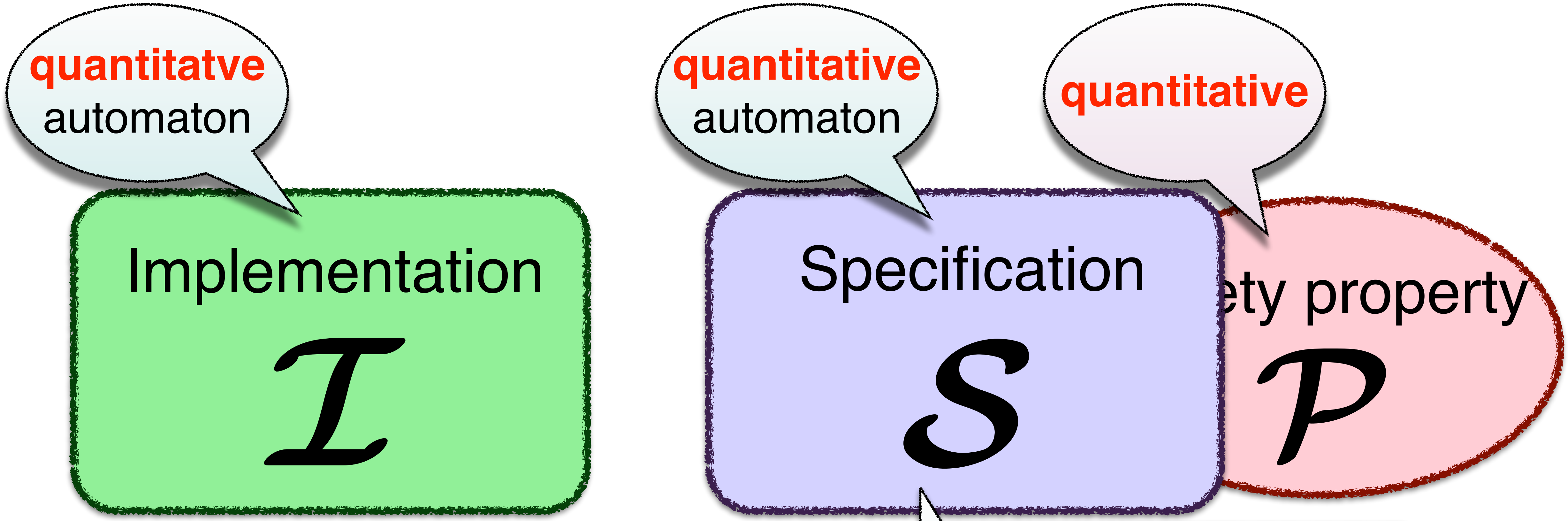
Obviously $\mathcal{S} \models \mathcal{P}$

Aim : Prove $\mathcal{I} \models \mathcal{P}$

$$\Uparrow$$

Prove $\mathrm{Lang}(\mathcal{I}) \subseteq \mathrm{Lang}(\mathcal{S})$ (**quantitative** language inclusion)

Simulation-based Verification for Quantitative Systems



Obviously $\mathcal{S} \models \mathcal{P}$

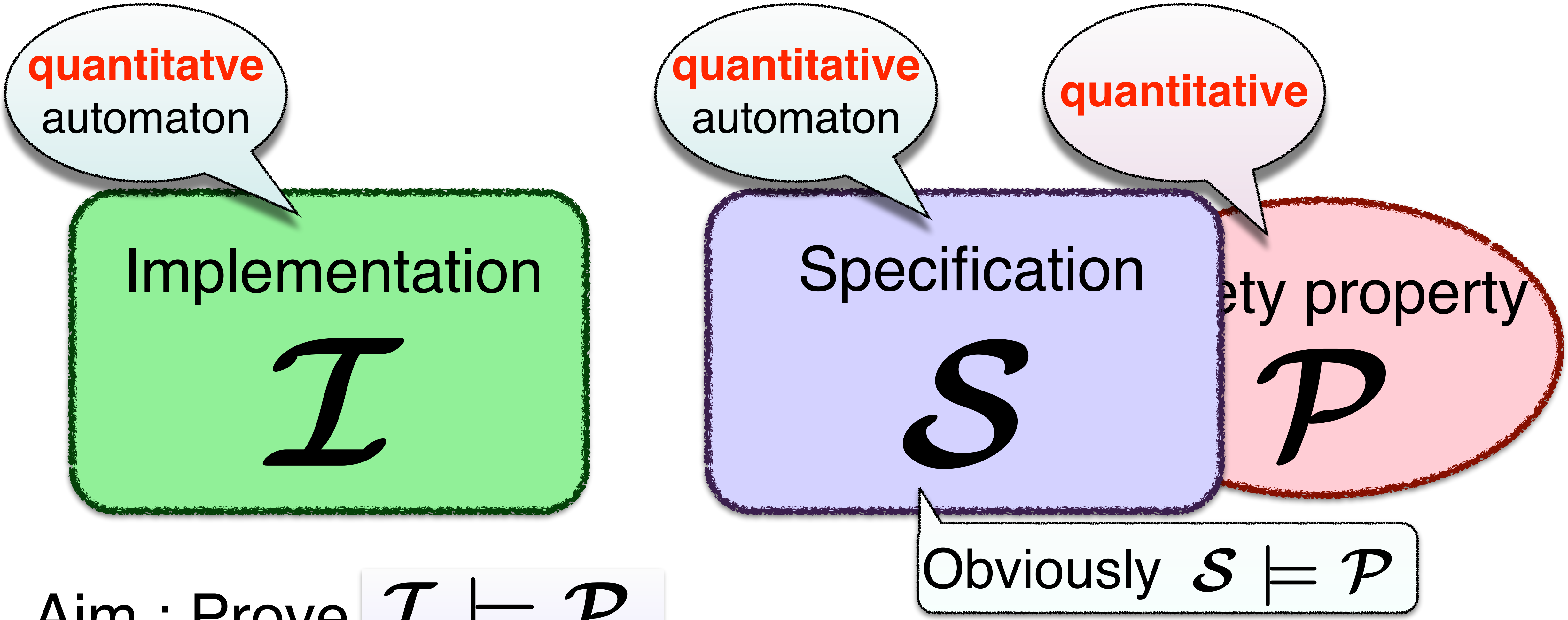
Aim : Prove $\mathcal{I} \models \mathcal{P}$

$\Uparrow$

Prove $\underline{\mathrm{Lang}(\mathcal{I}) \subseteq \mathrm{Lang}(\mathcal{S})}$ (**quantitative** language inclusion)

Hard to prove

Simulation-based Verification for Quantitative Systems



quantitatve automaton

Implementation

$\mathcal{I}$

quantitative automaton

Specification

$\mathcal{S}$

quantitative

ety property

$\mathcal{P}$

Obviously $\mathcal{S} \models \mathcal{P}$

Aim : Prove $\mathcal{I} \models \mathcal{P}$

$\Uparrow$

Prove $\underline{\mathrm{Lang}(\mathcal{I}) \subseteq \mathrm{Lang}(\mathcal{S})}$ (**quantitative** language inclusion)

Hard to prove → Quantitative simulation

Examples of Quantitative Simulation

- For $\mathcal{S}_{+,\times}$-weighted automata (probabilistic system)
  - Simulation by Jonsson & Larsen (1991)

- For $\mathcal{S}_{\mathbf{max},+}$ -weighted automata (system with cost)
  - Simulation by Chatterjee et al. (2010)

Our Result

- We defined **matrix simulation** for **semiring-weighted automaton**

Our Result

- We defined **matrix simulation** for

semiring-weighted automaton

Model for various quantitative systems

e.g. probability, cost, reward, …

Our Result

- We defined **matrix simulation** for

semiring-weighted automaton

Model for various quantitative systems

e.g. probability, cost, reward, …

➡ **Generality**

Our Result

- We defined **matrix simulation** for **semiring-weighted automaton**

Model for various quantitative systems

e.g. probability, cost, reward, …

→ **Generality**

Matrix that satisfies some inequalities on certain semiring

$$\begin{aligned} \alpha_{\mathcal{A}} &\sqsubseteq \alpha_{\mathcal{B}} X \\ X \cdot M_{\mathcal{A}}(a) &\sqsubseteq M_{\mathcal{B}}(a) \cdot X \quad (\forall a \in \Sigma) \\ X \beta_{\mathcal{A}} &\sqsubseteq \beta_{\mathcal{B}} \end{aligned}$$

$$\begin{aligned} \alpha_{\mathcal{A}} X &\sqsubseteq \alpha_{\mathcal{B}} \\ M_{\mathcal{A}}(a) \cdot X &\sqsubseteq X \cdot M_{\mathcal{B}}(a) \quad (\forall a \in \Sigma) \\ \beta_{\mathcal{A}} &\sqsubseteq X \beta_{\mathcal{B}} \end{aligned}$$

Our Result

- We defined **matrix simulation** for **semiring-weighted automaton**

Model for various quantitative systems

e.g. probability, cost, reward, …

→ **Generality**

Matrix that satisfies some inequalities on certain semiring

$$\begin{aligned} \alpha_{\mathcal{A}} &\sqsubseteq \alpha_{\mathcal{B}} X \\ X \cdot M_{\mathcal{A}}(a) &\sqsubseteq M_{\mathcal{B}}(a) \cdot X \quad (\forall a \in \Sigma) \\ X \beta_{\mathcal{A}} &\sqsubseteq \beta_{\mathcal{B}} \end{aligned}$$

$$\begin{aligned} \alpha_{\mathcal{A}} X &\sqsubseteq \alpha_{\mathcal{B}} \\ M_{\mathcal{A}}(a) \cdot X &\sqsubseteq X \cdot M_{\mathcal{B}}(a) \quad (\forall a \in \Sigma) \\ \beta_{\mathcal{A}} &\sqsubseteq X \beta_{\mathcal{B}} \end{aligned}$$

→ **Practicality**

Our Result

- We defined **matrix simulation** for **semiring-weighted automaton**

Weak point: **Sound** but **not complete**

Our Result

- We defined **matrix simulation** for **semiring-weighted automaton**

Weak point: **Sound** but **not complete**

- Introduce **fwd./ bwd. partial execution**
  - Transformation of automaton that produces matrix simulation

Our Result

- We defined **matrix simulation** for **semiring-weighted automaton**

Weak point: **Sound** but **not complete**

- Introduce **fwd./ bwd. partial execution**
  - Transformation of automaton that produces matrix simulation

- Proof-of-concept implementation and experiment

Overview

1. Matrix Simulation
    - Motivation
    - Semiring-Weighted Automaton and Matrix Simulation
    - Origin: from Theory of Coalgebra
2. Partial Execution (to be More “Complete”)
3. Specific Examples
    - Example 1 : $\boldsymbol{\mathcal{S}}_{+,\times}$ -weighted Automaton
    - Example 2 : $\boldsymbol{\mathcal{S}}_{\mathbf{max},+}$ -weighted Automaton
4. Conclusion and Future Works

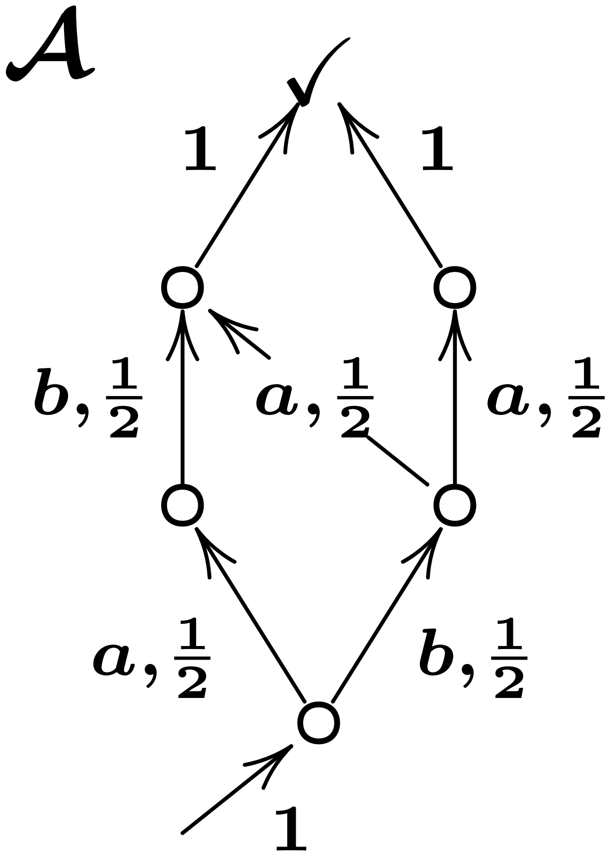
Overview

1. Matrix Simulation
    - Motivation
    - Semiring-Weighted Automaton and Matrix Simulation
    - Origin: from Theory of Coalgebra
2. Partial Execution (to be More "Complete")
3. Specific Examples
    - Example 1 : $\boldsymbol{\mathcal{S}}_{+,\times}$ -weighted Automaton
    - Example 2 : $\boldsymbol{\mathcal{S}}_{\mathbf{max},+}$ -weighted Automaton
4. Conclusion and Future Works

Example of Semiring-Weighted Automaton

- **Semiring Weighted Automaton:** Automaton weighted with values in semiring

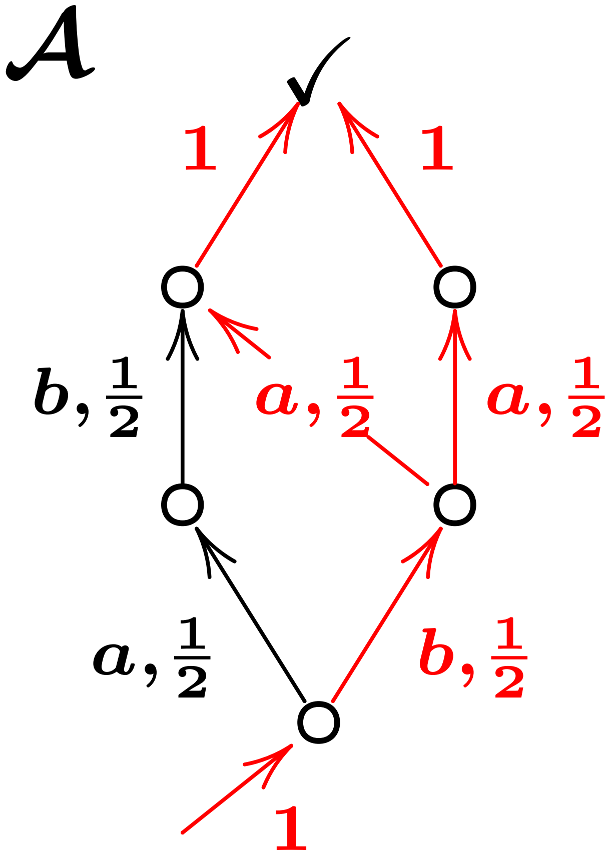
Various semirings for various systems



Example of Semiring-Weighted Automaton

- **Semiring Weighted Automaton:** Automaton weighted with values in semiring

Various semirings for various systems

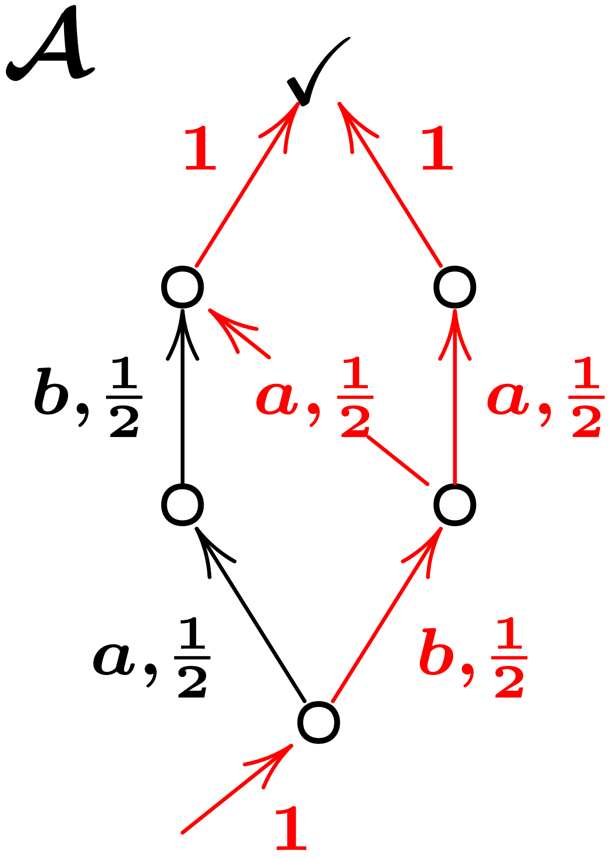


$$\mathrm{Lang}(\mathcal{A})(ba) = \Bigg\{$$

Example of Semiring-Weighted Automaton

- **Semiring Weighted Automaton:** Automaton weighted with values in semiring

Various semirings for various systems



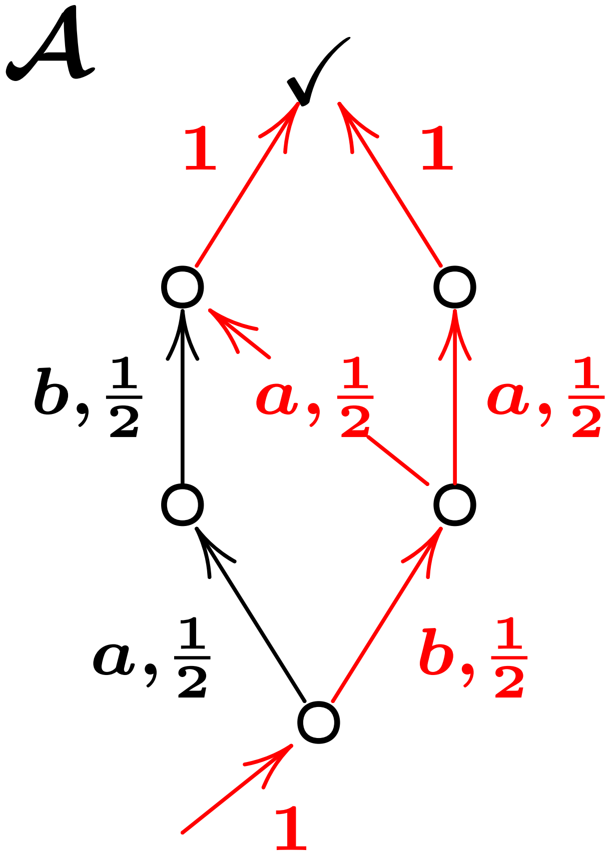
Weight: **probability**

$$\mathrm{Lang}(\mathcal{A})(ba) = \Bigg\{$$

Example of Semiring-Weighted Automaton

- **Semiring Weighted Automaton:** Automaton weighted with values in semiring

Various semirings for various systems



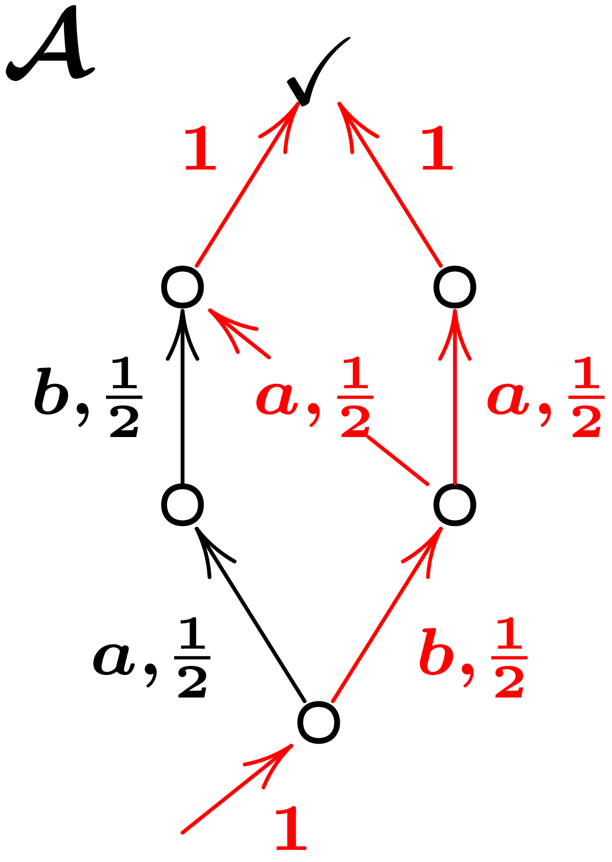
Weight: **probability**

$$\mathrm{Lang}(\mathcal{A})(ba) = \begin{cases} 1 \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot 1 + 1 \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot 1 \end{cases}$$

Example of Semiring-Weighted Automaton

- Semiring Weighted Automaton: Automaton weighted with values in semiring

Various semirings for various systems



Weight: **probability**

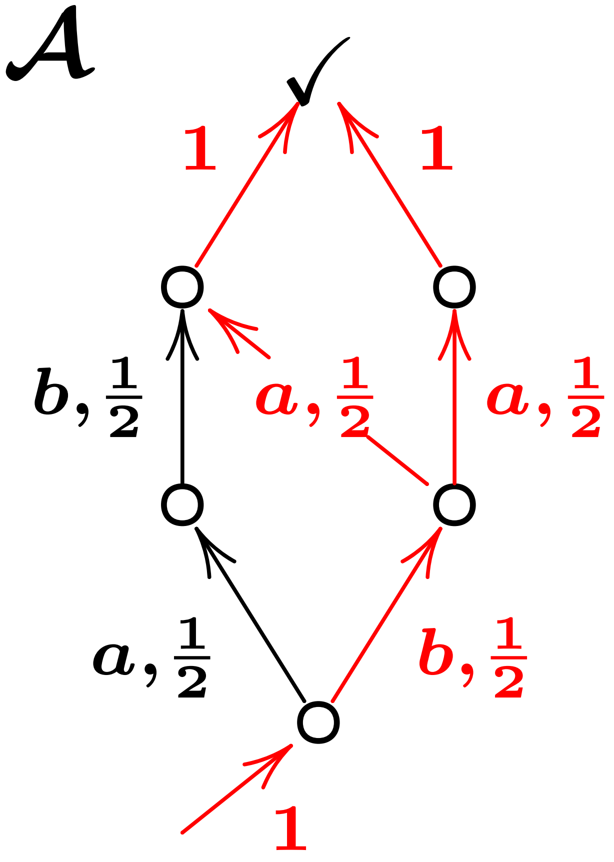
$$\mathrm{Lang}(\mathcal{A})(ba) = \begin{cases} 1 \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot 1 + 1 \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot 1 \end{cases}$$

Weight: (worst case) **resource consumption**

Example of Semiring-Weighted Automaton

- Semiring Weighted Automaton: Automaton weighted with values in semiring

Various semirings for various systems



Weight: **probability**

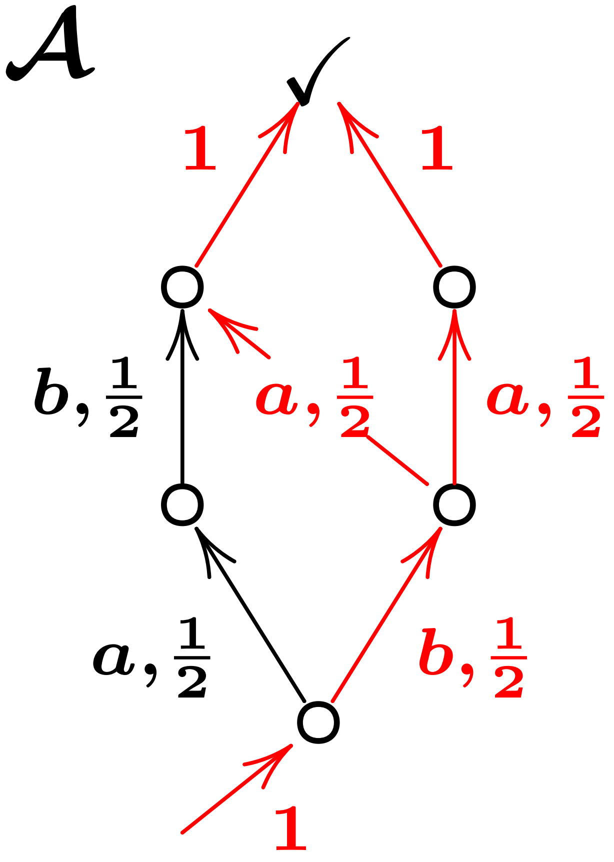
$$\mathrm{Lang}(\mathcal{A})(ba) = \left\{ \begin{array}{l} 1 \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot 1 + 1 \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot 1 \\ \max(1 + \frac{1}{2} + \frac{1}{2} + 1, 1 + \frac{1}{2} + \frac{1}{2} + 1) \end{array} \right.$$

Weight: (worst case) **resource consumption**

Example of Semiring-Weighted Automaton

- **Semiring Weighted Automaton:** Automaton weighted with values in semiring

Various semirings for various systems



$$\mathcal{S}_{+,\times} = ([0,\infty], +, 0, \times, 1, \leq)$$

Weight: **probability**

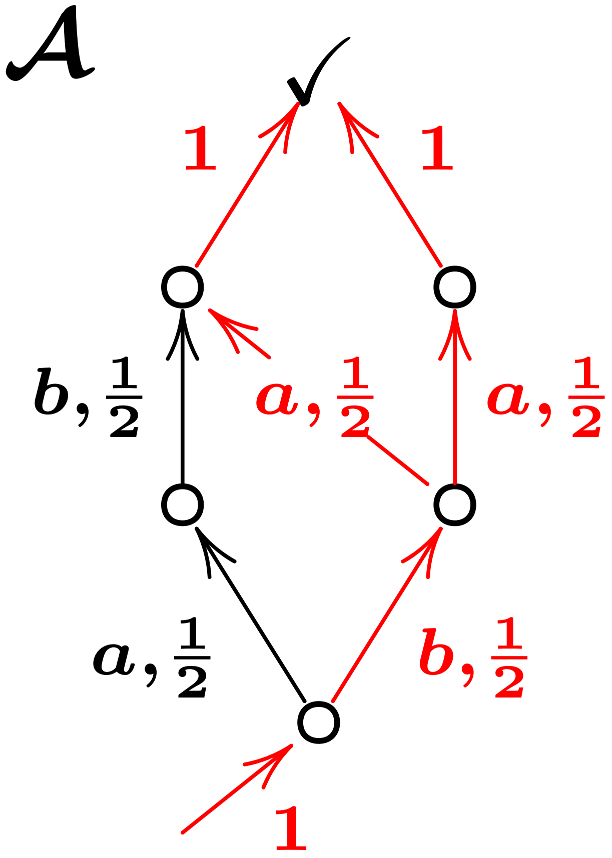
$$\mathrm{Lang}(\mathcal{A})(ba) = \begin{cases} 1 \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot 1 + 1 \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot 1 \\ \max(1 + \frac{1}{2} + \frac{1}{2} + 1, 1 + \frac{1}{2} + \frac{1}{2} + 1) \end{cases}$$

Weight: (worst case) **resource consumption**

Example of Semiring-Weighted Automaton

- **Semiring Weighted Automaton:** Automaton weighted with values in semiring

Various semirings for various systems



$\mathcal{S}_{+,\times} = ([0,\infty], +, 0, \times, 1, \leq)$

Weight: **probability**

$$\mathrm{Lang}(\mathcal{A})(ba) = \begin{cases} 1 \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot 1 + 1 \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot 1 \\ \max(1 + \frac{1}{2} + \frac{1}{2} + 1, 1 + \frac{1}{2} + \frac{1}{2} + 1) \end{cases}$$

Weight: (worst case) **resource consumption**

$\mathcal{S}_{\max,+} = ([-\infty,\infty], \max, -\infty, +, 0, \leq)$

Formal Definition of Weighted Automaton

Def : For a commutative cppo-semiring $\mathcal{S} = (S, +_{\mathcal{S}}, 0_{\mathcal{S}}, \times_{\mathcal{S}}, 1_{\mathcal{S}}, \sqsubseteq)$,

$\mathcal{S}$ -weighted automaton $\mathcal{A} = (Q, \Sigma, M, \alpha, \beta)$ consists of

- a state space Q
- alphabet Σ
- transition matrices $M(a) \in \mathcal{S}^{Q \times Q}$
- a initial (row) vector $\alpha \in \mathcal{S}^{Q}$
- a final (column) vector $\beta \in \mathcal{S}^{Q}$

Formal Definition of Weighted Automaton

Def : For a commutative cppo-semiring $\mathcal{S} = (S, +_{\mathcal{S}}, 0_{\mathcal{S}}, \times_{\mathcal{S}}, 1_{\mathcal{S}}, \sqsubseteq)$,

$\mathcal{S}$ -weighted automaton $\mathcal{A} = (Q, \Sigma, M, \alpha, \beta)$ consists of

- a state space Q
- alphabet Σ
- transition matrices $M(a) \in \mathcal{S}^{Q \times Q}$
- a initial (row) vector $\alpha \in \mathcal{S}^{Q}$
- a final (column) vector $\beta \in \mathcal{S}^{Q}$

- Language of $\mathcal{A}$: $\mathrm{Lang}(\mathcal{A})$

$$\mathrm{Lang}(\mathcal{A})(a_0 a_1 \ldots a_n) := \boxed{\quad \alpha \quad} \ \boxed{M(a_0)} \ \boxed{M(a_1)} \ \ldots \ \boxed{M(a_n)} \ \boxed{\beta}$$

Formal Definition of Weighted Automaton

Def : For a commutative cppo-semiring $\mathcal{S} = (S, +_{\mathcal{S}}, 0_{\mathcal{S}}, \times_{\mathcal{S}}, 1_{\mathcal{S}}, \sqsubseteq)$,

$\mathcal{S}$ -weighted automaton $\mathcal{A} = (Q, \Sigma, M, \alpha, \beta)$ consists of

- a state space Q
- alphabet Σ
- transition matrices $M(a) \in \mathcal{S}^{Q \times Q}$
- a initial (row) vector $\alpha \in \mathcal{S}^{Q}$
- a final (column) vector $\beta \in \mathcal{S}^{Q}$

- Language of $\mathcal{A}$: $\mathrm{Lang}(\mathcal{A})$

$$\mathrm{Lang}(\mathcal{A})(a_0 a_1 \dots a_n) := \boxed{\quad \alpha \quad} \boxed{M(a_0)} \boxed{M(a_1)} \dots \boxed{M(a_n)} \boxed{\beta}$$

- Language inclusion :

$$\mathrm{Lang}(\mathcal{A}) \sqsubseteq \mathrm{Lang}(\mathcal{B}) \overset{\text{def}}{\Leftrightarrow} \forall w.\ \mathrm{Lang}(\mathcal{A})(w) \sqsubseteq \mathrm{Lang}(\mathcal{B})(w)$$

Formal Definition of Weighted Automaton

Def : For a commutative cppo-semiring $\mathcal{S} = (S, +_{\mathcal{S}}, 0_{\mathcal{S}}, \times_{\mathcal{S}}, 1_{\mathcal{S}}, \sqsubseteq)$,

$\mathcal{S}$ -weighted automaton $\mathcal{A} = (Q, \Sigma, M, \alpha, \beta)$ consists of

- a state space Q
- alphabet Σ
- transition matrices $M(a) \in \mathcal{S}^{Q \times Q}$
- a initial (row) vector $\alpha \in \mathcal{S}^{Q}$
- a final (column) vector $\beta \in \mathcal{S}^{Q}$

- Language of $\mathcal{A}$: $\mathrm{Lang}(\mathcal{A})$

$$\mathrm{Lang}(\mathcal{A})(a_0 a_1 \ldots a_n) := \boxed{\quad \alpha \quad} \boxed{M(a_0)} \boxed{M(a_1)} \ldots \boxed{M(a_n)} \boxed{\beta}$$

- Language inclusion :

$$\mathrm{Lang}(\mathcal{A}) \sqsubseteq \mathrm{Lang}(\mathcal{B}) \overset{\text{def}}{\Leftrightarrow} \boxed{\forall w.\ \mathrm{Lang}(\mathcal{A})(w) \sqsubseteq \mathrm{Lang}(\mathcal{B})(w)}$$

What we want to prove

Matrix Simulation

Def : For weighted automata $\mathcal{A} = (Q_{\mathcal{A}}, \Sigma, M_{\mathcal{A}}, \alpha_{\mathcal{A}}, \beta_{\mathcal{A}})$ and $\mathcal{B} = (Q_{\mathcal{B}}, \Sigma, M_{\mathcal{B}}, \alpha_{\mathcal{B}}, \beta_{\mathcal{B}})$,

- a forward simulation matrix is $X \in S^{Q_{\mathcal{B}} \times Q_{\mathcal{A}}}$ such that

  $\alpha_{\mathcal{A}} \sqsubseteq \alpha_{\mathcal{B}} X \ , \quad X \cdot M_{\mathcal{A}}(a) \sqsubseteq M_{\mathcal{B}}(a) \cdot X \quad (\forall a \in \Sigma) \ , \quad \text{and} \quad X \beta_{\mathcal{A}} \sqsubseteq \beta_{\mathcal{B}}$

- a backward simulation matrix is $X \in S^{Q_{\mathcal{A}} \times Q_{\mathcal{B}}}$ such that

  $\alpha_{\mathcal{A}} X \sqsubseteq \alpha_{\mathcal{B}} \ , \quad M_{\mathcal{A}}(a) \cdot X \sqsubseteq X \cdot M_{\mathcal{B}}(a) \quad (\forall a \in \Sigma) \ , \quad \text{and} \quad \beta_{\mathcal{A}} \sqsubseteq X \beta_{\mathcal{B}}$

- Two types: **Forward** and **Backward**
- Both defined as **matrix** satisfying certain inequalities

Matrix Simulation

Def : For weighted automata $\mathcal{A} = (Q_{\mathcal{A}}, \Sigma, M_{\mathcal{A}}, \alpha_{\mathcal{A}}, \beta_{\mathcal{A}})$ and $\mathcal{B} = (Q_{\mathcal{B}}, \Sigma, M_{\mathcal{B}}, \alpha_{\mathcal{B}}, \beta_{\mathcal{B}})$,

- a forward simulation matrix is $X \in S^{Q_{\mathcal{B}} \times Q_{\mathcal{A}}}$ such that

  $\alpha_{\mathcal{A}} \sqsubseteq \alpha_{\mathcal{B}} X$, $X \cdot M_{\mathcal{A}}(a) \sqsubseteq M_{\mathcal{B}}(a) \cdot X$ $(\forall a \in \Sigma)$, and $X \beta_{\mathcal{A}} \sqsubseteq \beta_{\mathcal{B}}$

- a backward simulation matrix is $X \in S^{Q_{\mathcal{A}} \times Q_{\mathcal{B}}}$ such that

  $\alpha_{\mathcal{A}} X \sqsubseteq \alpha_{\mathcal{B}}$, $M_{\mathcal{A}}(a) \cdot X \sqsubseteq X \cdot M_{\mathcal{B}}(a)$ $(\forall a \in \Sigma)$, and $\beta_{\mathcal{A}} \sqsubseteq X \beta_{\mathcal{B}}$

- Two types: **Forward** and **Backward**
- Both defined as **matrix** satisfying certain inequalities

linear inequalities

→ A lot of existing research

Matrix Simulation

Def : For weighted automata $\mathcal{A} = (Q_\mathcal{A}, \Sigma, M_\mathcal{A}, \alpha_\mathcal{A}, \beta_\mathcal{A})$ and $\mathcal{B} = (Q_\mathcal{B}, \Sigma, M_\mathcal{B}, \alpha_\mathcal{B}, \beta_\mathcal{B})$,

- a forward simulation matrix is $X \in S^{Q_\mathcal{B} \times Q_\mathcal{A}}$ such that

  $$\alpha_\mathcal{A} \sqsubseteq \alpha_\mathcal{B} X \;,\quad X \cdot M_\mathcal{A}(a) \sqsubseteq M_\mathcal{B}(a) \cdot X \quad (\forall a \in \Sigma) \;,\quad \text{and} \quad X\beta_\mathcal{A} \sqsubseteq \beta_\mathcal{B}$$

- a backward simulation matrix is $X \in S^{Q_\mathcal{A} \times Q_\mathcal{B}}$ such that

  $$\alpha_\mathcal{A} X \sqsubseteq \alpha_\mathcal{B} \;,\quad M_\mathcal{A}(a) \cdot X \sqsubseteq X \cdot M_\mathcal{B}(a) \quad (\forall a \in \Sigma) \;,\quad \text{and} \quad \beta_\mathcal{A} \sqsubseteq X\beta_\mathcal{B}$$

Thm (Soundness) :

A fwd. simulation matrix exists $\Longrightarrow$ $\forall w.\; L(\mathcal{A})(w) \sqsubseteq L(\mathcal{B})(w)$

A bwd. simulation matrix exists $\Longrightarrow$ $\forall w.\; L(\mathcal{A})(w) \sqsubseteq L(\mathcal{B})(w)$

Overview

1. Matrix Simulation
    - Motivation
    - Semiring-Weighted Automaton and Matrix Simulation
    - Origin: from Theory of Coalgebra
2. Partial Execution (to be More “Complete”)
3. Specific Examples
    - Example 1 : $\boldsymbol{\mathcal{S}}_{+,\times}$ -weighted Automaton
    - Example 2 : $\boldsymbol{\mathcal{S}}_{\mathbf{max},+}$ -weighted Automaton
4. Conclusion and Future Works

Theory behind Matrix Simulation

Matrix simulation

$\mathcal{S}$-weighted automaton

$$\alpha_{\mathcal{A}} \sqsubseteq \alpha_{\mathcal{B}} X \ ,$$
$$X \cdot M_{\mathcal{A}}(a) \sqsubseteq M_{\mathcal{B}}(a) \cdot X \quad (\forall a \in \Sigma) \ ,$$
$$X \beta_{\mathcal{A}} \sqsubseteq \beta_{\mathcal{B}}$$

- Matrix simulation is obtained via **Kleisli simulation** **[Hasuo, 2006]**
  - **Kleisli Simulation** :

    Categorical generalization of simulation by Lynch & Vaandrager (1995)
  - Using theory of coalgebra, we can prove soundness in general

Theory behind Matrix Simulation

Systems represented as coalgebra

$$\begin{tikzcd}
FX & & FY \arrow[ll, "Ff"', "|" marking] \\
 & \sqsubseteq & \\
X \arrow[uu, "c"', "|" marking] & & Y \arrow[ll, "f", "|" marking] \arrow[uu, "d"', "|" marking] \\
 & \sqsubseteq & \\
 & \{\bullet\} \arrow[uul, bend left, "s", "|" marking] \arrow[uur, bend right, "t"', "|" marking] &
\end{tikzcd}$$

Kleisli simulation by Hasuo (2006)

specialize

Matrix simulation

$\mathcal{S}$-weighted automaton

$$\alpha_{\mathcal{A}} \sqsubseteq \alpha_{\mathcal{B}} X \ ,$$

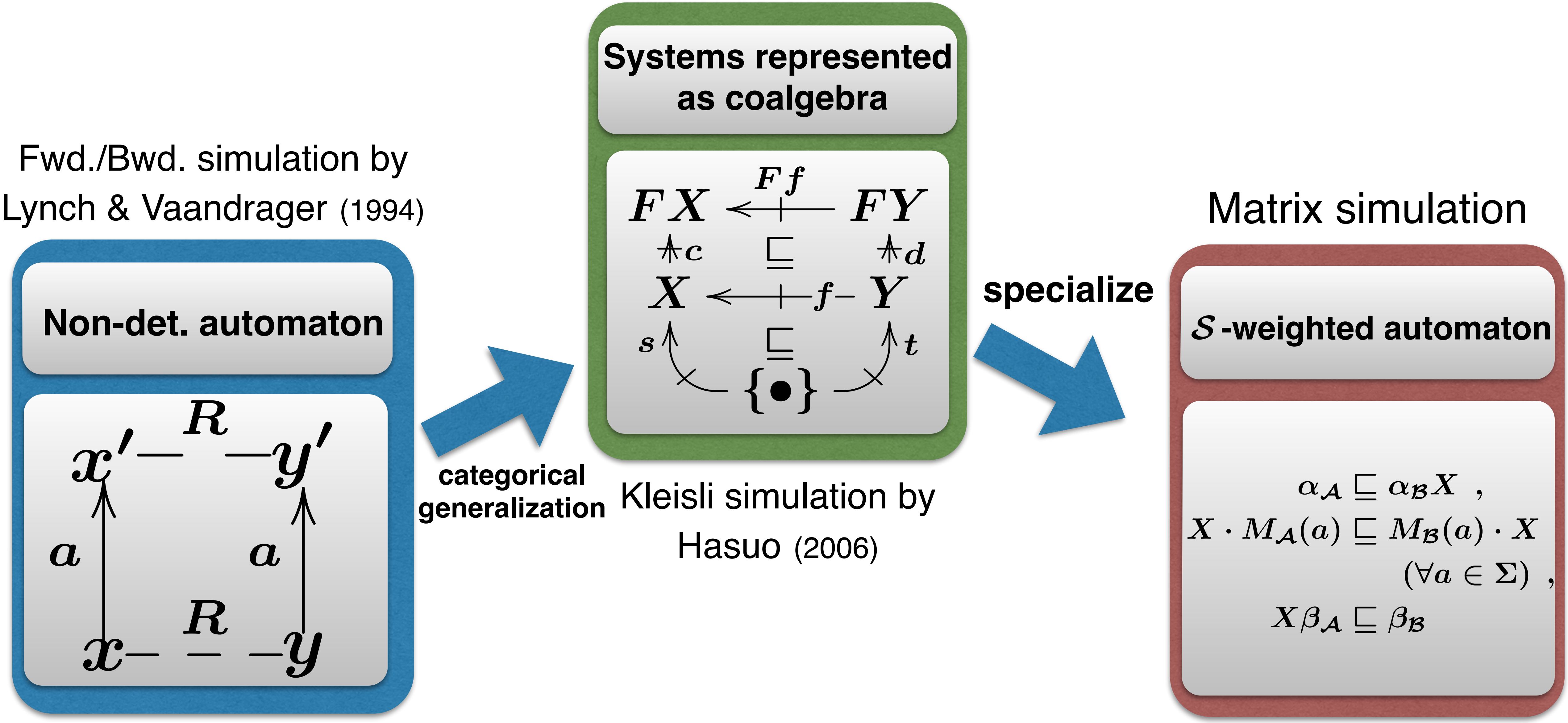
$$X \cdot M_{\mathcal{A}}(a) \sqsubseteq M_{\mathcal{B}}(a) \cdot X \quad (\forall a \in \Sigma) \ ,$$

$$X \beta_{\mathcal{A}} \sqsubseteq \beta_{\mathcal{B}}$$

- Matrix simulation is obtained via **Kleisli simulation [Hasuo, 2006]**
  - **Kleisli Simulation** :

    Categorical generalization of simulation by Lynch & Vaandrager (1995)
  - Using theory of coalgebra, we can prove soundness in general

Theory behind Matrix Simulation



Fwd./Bwd. simulation by Lynch & Vaandrager (1994)

Non-det. automaton

$$\begin{tikzcd}
x' \arrow[r, dash, "R"] & y' \\
x \arrow[u, "a"] \arrow[r, dash, dashed, "R"] & y \arrow[u, "a"']
\end{tikzcd}$$

categorical generalization

Systems represented as coalgebra

$$\begin{tikzcd}
FX & & FY \arrow[ll, "Ff"'] \\
& \sqsubseteq & \\
X \arrow[uu, "c"] & & Y \arrow[ll, "f"] \arrow[uu, "d"'] \\
& \sqsubseteq & \\
& \{\bullet\} \arrow[uul, bend left, "s"] \arrow[uur, bend right, "t"'] &
\end{tikzcd}$$

Kleisli simulation by Hasuo (2006)

specialize

Matrix simulation

$\mathcal{S}$-weighted automaton

$$\alpha_{\mathcal{A}} \sqsubseteq \alpha_{\mathcal{B}} X \ ,$$

$$X \cdot M_{\mathcal{A}}(a) \sqsubseteq M_{\mathcal{B}}(a) \cdot X \quad (\forall a \in \Sigma) \ ,$$

$$X \beta_{\mathcal{A}} \sqsubseteq \beta_{\mathcal{B}}$$

- Matrix simulation is obtained via **Kleisli simulation** [Hasuo, 2006]
  - **Kleisli Simulation** :

    Categorical generalization of simulation by Lynch & Vaandrager (1995)
  - Using theory of coalgebra, we can prove soundness in general

Coalgebraic Modeling of Transition System

- Represented system as coalgebra $c : X \to TFX$

Coalgebraic Modeling of Transition System

- Represented system as coalgebra $c : X \to \enclose{circle}{T} F X$

T : Monad representing branching type

Coalgebraic Modeling of Transition System

- Represented system as coalgebra $c : X \to TFX$

T : Monad representing branching type

F : Functor representing transition type

Coalgebraic Modeling of Transition System

- Represented system as coalgebra $c : X \to TFX$

T : Monad representing branching type

e.g.

$T = \mathcal{P}$ (powerset monad) : non-deterministic system

$T = \mathcal{D}$ (subdistribution monad) : probabilistic system

$T = \mathcal{M}_{\mathcal{S}}$ (multiset monad) : $\mathcal{S}$-weighted system

F : Functor representing transition type

e.g.

$F = 1 + \Sigma \times (\_)$: automaton for finite-length word

$F = 1 + \Sigma \times (\_) \times (\_)$: automaton for finite-depth tree

- **Various** choice for T and F

➡ We can represent **various** systems

Coalgebraic Modeling of Transition System

- Represented system as coalgebra $c : X \to TFX$

T : Monad representing branching type

e.g.

$T = \mathcal{P}$ (powerset monad) : non-deterministic system

$T = \mathcal{D}$ (subdistribution monad) : probabilistic system

$T = \mathcal{M}_{\mathcal{S}}$ (multiset monad) : $\mathcal{S}$-weighted system

Our setting

F : Functor representing transition type

e.g.

$F = 1 + \Sigma \times (\_)$: automaton for finite-length word

$F = 1 + \Sigma \times (\_) \times (\_)$: automaton for finite-depth tree

- **Various** choice for T and F

⇒ We can represent **various** systems

Transition System as Kleisli Arrow

- Represented system as coalgebra $\boldsymbol{c : X \rightarrow TFX}$

- This arrow can be regarded as Kleisli arrow

$$\begin{array}{c} \boldsymbol{c : X \rightarrow TFX} \quad \text{in } \mathbf{Set} \\ \hline\hline \boldsymbol{c : X \mathrel{\mapstochar\rightarrow} FX} \quad \text{in } \mathcal{K}\ell(\boldsymbol{T}) \end{array}$$

Def: Kleisli arrow

$$\begin{array}{c} \boldsymbol{f : X \rightarrow TY} \quad \text{in } \mathbf{Set} \\ \hline\hline \boldsymbol{f : X \mathrel{\mapstochar\rightarrow} Y} \quad \text{in } \mathcal{K}\ell(\boldsymbol{T}) \end{array}$$

Kleisli Simulation [Hasuo 2006]

- Forward / Backward Kleisli simulation is
  a Kleisli arrow satisfying a certain diagram

$$\begin{tikzcd}
FX & & FY \arrow[ll, "Ff"', "\shortmid" marking] \\
& \sqsubseteq & \\
X \arrow[uu, "c"', "\shortmid" marking] & & Y \arrow[ll, "f", "\shortmid" marking] \arrow[uu, "d", "\shortmid" marking] \\
& \sqsubseteq & \\
& \{\bullet\} \arrow[uul, bend left, "s", "\shortmid" marking] \arrow[uur, bend right, "t"', "\shortmid" marking] &
\end{tikzcd}$$

Forward Kleisli arrow f

$$\begin{tikzcd}
FX \arrow[rr, "Fb", "\shortmid" marking] & & FY \\
& \sqsubseteq & \\
X \arrow[uu, "c"', "\shortmid" marking] \arrow[rr, "b", "\shortmid" marking] & & Y \arrow[uu, "d", "\shortmid" marking] \\
& \sqsubseteq & \\
& \{\bullet\} \arrow[uul, bend left, "s", "\shortmid" marking] \arrow[uur, bend right, "t"', "\shortmid" marking] &
\end{tikzcd}$$

Backward Kleisli arrow b

Kleisli Simulation [Hasuo 2006]

- Forward / Backward Kleisli simulation is
  a Kleisli arrow satisfying a certain diagram

$$\begin{tikzcd}
FX & & FY \arrow[ll, "Ff"', "|" marking] \\
X \arrow[u, "c"', "|" marking] & & Y \arrow[ll, "f", "|" marking] \arrow[u, "d"', "|" marking] \\
& \{\bullet\} \arrow[ul, bend left, "s", "|" marking] \arrow[ur, bend right, "t"', "|" marking] &
\arrow[from=1-1, to=2-3, phantom, "\sqsubseteq"]
\arrow[from=2-2, to=3-2, phantom, "\sqsubseteq"]
\end{tikzcd}$$

Forward Kleisli arrow f

$$\begin{tikzcd}
FX \arrow[rr, "Fb", "|" marking] & & FY \\
X \arrow[u, "c"', "|" marking] \arrow[rr, "b", "|" marking] & & Y \arrow[u, "d"', "|" marking] \\
& \{\bullet\} \arrow[ul, bend left, "s", "|" marking] \arrow[ur, bend right, "t"', "|" marking] &
\arrow[from=1-1, to=2-3, phantom, "\sqsubseteq"]
\arrow[from=2-2, to=3-2, phantom, "\sqsubseteq"]
\end{tikzcd}$$

Backward Kleisli arrow b

Thm (Soundness) :

Fwd. Kleisli simulation exists $\Rrightarrow$

Bwd. Kleisli simulation exists $\Rrightarrow$

$$\mathsf{tr}(c) \odot s \sqsubseteq \mathsf{tr}(d) \odot t$$

(i.e. trace inclusion)

Kleisli Simulation to Matrix Simulation

$$\begin{tikzcd}
\boldsymbol{FX} & & \boldsymbol{FY} \arrow[ll, "\boldsymbol{Ff}"', "|"{marking}] \\
\boldsymbol{X} \arrow[u, "\boldsymbol{c}"', "|"{marking}] & & \boldsymbol{Y} \arrow[ll, "\boldsymbol{f}", "|"{marking}] \arrow[u, "\boldsymbol{d}"', "|"{marking}] \\
& \{\bullet\} \arrow[ul, bend left, "\boldsymbol{s}", "|"{marking}] \arrow[ur, bend right, "\boldsymbol{t}"', "|"{marking}] &
\arrow[from=1-2, to=2-2, phantom, "\sqsubseteq"]
\arrow[from=2-2, to=3-2, phantom, "\sqsubseteq"]
\end{tikzcd}$$

Forward Kleisli arrow $\boldsymbol{f}$

$$\begin{tikzcd}
\boldsymbol{FX} \arrow[rr, "\boldsymbol{Fb}", "|"{marking}] & & \boldsymbol{FY} \\
\boldsymbol{X} \arrow[u, "\boldsymbol{c}"', "|"{marking}] \arrow[rr, "\boldsymbol{b}", "|"{marking}] & & \boldsymbol{Y} \arrow[u, "\boldsymbol{d}"', "|"{marking}] \\
& \{\bullet\} \arrow[ul, bend left, "\boldsymbol{s}", "|"{marking}] \arrow[ur, bend right, "\boldsymbol{t}"', "|"{marking}] &
\arrow[from=1-2, to=2-2, phantom, "\sqsubseteq"]
\arrow[from=2-2, to=3-2, phantom, "\sqsubseteq"]
\end{tikzcd}$$

Backward Kleisli arrow $\boldsymbol{b}$

Kleisli Simulation to Matrix Simulation

$$\begin{tikzcd}
FX & & FY \arrow[ll, "\shortmid" marking, "Ff"'] \\
& \sqsubseteq & \\
X \arrow[uu, "\shortmid" marking, "c"'] & & Y \arrow[ll, "\shortmid" marking, "f"] \arrow[uu, "\shortmid" marking, "d"'] \\
& \sqsubseteq & \\
& \{\bullet\} \arrow[uul, bend left, "\shortmid" marking, "s"] \arrow[uur, bend right, "\shortmid" marking, "t"'] &
\end{tikzcd}$$

Forward Kleisli arrow f

$$\begin{tikzcd}
FX \arrow[rr, "\shortmid" marking, "Fb"] & & FY \\
& \sqsubseteq & \\
X \arrow[uu, "\shortmid" marking, "c"'] \arrow[rr, "\shortmid" marking, "b"] & & Y \arrow[uu, "\shortmid" marking, "d"'] \\
& \sqsubseteq & \\
& \{\bullet\} \arrow[uul, bend left, "\shortmid" marking, "s"] \arrow[uur, bend right, "\shortmid" marking, "t"'] &
\end{tikzcd}$$

Backward Kleisli arrow b

- For $\mathcal{S}$-weighted automata,

  $T = \mathcal{M}_{\mathcal{S}}$ (multiset monad) : $\mathcal{S}$-weighted system

Kleisli Simulation to Matrix Simulation

$$\begin{tikzcd}
FX & & FY \arrow[ll, "Ff"', "\shortmid" marking] \\
& \sqsubseteq & \\
X \arrow[uu, "c"', "\shortmid" marking] & & Y \arrow[ll, "f", "\shortmid" marking] \arrow[uu, "d"', "\shortmid" marking] \\
& \sqsubseteq & \\
& \{\bullet\} \arrow[uul, bend left, "s", "\shortmid" marking] \arrow[uur, bend right, "t"', "\shortmid" marking] &
\end{tikzcd}$$

Forward Kleisli arrow f

$$\begin{tikzcd}
FX \arrow[rr, "Fb", "\shortmid" marking] & & FY \\
& \sqsubseteq & \\
X \arrow[uu, "c"', "\shortmid" marking] \arrow[rr, "b", "\shortmid" marking] & & Y \arrow[uu, "d"', "\shortmid" marking] \\
& \sqsubseteq & \\
& \{\bullet\} \arrow[uul, bend left, "s", "\shortmid" marking] \arrow[uur, bend right, "t"', "\shortmid" marking] &
\end{tikzcd}$$

Backward Kleisli arrow b

- For $\mathcal{S}$-weighted automata,

$T = \mathcal{M}_{\mathcal{S}}$ (multiset monad) : $\mathcal{S}$-weighted system

Kleisli arrows in $\mathcal{K}\ell(\mathcal{M}_S)$

$f : X \nrightarrow \mathcal{M}_S Y$

Kleisli Simulation to Matrix Simulation

$$\begin{tikzcd}
FX & & FY \arrow[ll, "Ff"', "|" marking] \\
 & \sqsubseteq & \\
X \arrow[uu, "c"', "|" marking] & & Y \arrow[ll, "f", "|" marking] \arrow[uu, "d", "|" marking] \\
 & \sqsubseteq & \\
 & \{\bullet\} \arrow[uul, bend left, "s", "|" marking] \arrow[uur, bend right, "t"', "|" marking] &
\end{tikzcd}$$

Forward Kleisli arrow f

$$\begin{tikzcd}
FX \arrow[rr, "Fb", "|" marking] & & FY \\
 & \sqsubseteq & \\
X \arrow[uu, "c"', "|" marking] \arrow[rr, "b", "|" marking] & & Y \arrow[uu, "d", "|" marking] \\
 & \sqsubseteq & \\
 & \{\bullet\} \arrow[uul, bend left, "s", "|" marking] \arrow[uur, bend right, "t"', "|" marking] &
\end{tikzcd}$$

Backward Kleisli arrow b

- For $\mathcal{S}$-weighted automata,

$T = \mathcal{M}_{\mathcal{S}}$ (multiset monad) : $\mathcal{S}$-weighted system

Kleisli arrows in $\mathcal{K}\ell(\mathcal{M}_S)$

$$f : X \nrightarrow \mathcal{M}_S Y$$

Matrix on $\mathcal{S}$

$X_f \in \mathcal{S}^{A \times B}$

s.t. $(X_f)_{a,b} = f(a)(b)$

Kleisli Simulation to Matrix Simulation

$$\begin{tikzcd}
FX & & FY \arrow[ll, "|" marking, "Ff"'] \\
& \sqsubseteq & \\
X \arrow[uu, "c"'] & & Y \arrow[ll, "|" marking, "f"] \arrow[uu, "d"'] \\
& \sqsubseteq & \\
& \{\bullet\} \arrow[uul, bend left, "\times" marking, "s"] \arrow[uur, bend right, "\times" marking, "t"'] &
\end{tikzcd}$$

Forward Kleisli arrow f

$$\begin{tikzcd}
FX \arrow[rr, "|" marking, "Fb"] & & FY \\
& \sqsubseteq & \\
X \arrow[uu, "c"'] \arrow[rr, "|" marking, "b"] & & Y \arrow[uu, "d"'] \\
& \sqsubseteq & \\
& \{\bullet\} \arrow[uul, bend left, "\times" marking, "s"] \arrow[uur, bend right, "\times" marking, "t"'] &
\end{tikzcd}$$

Backward Kleisli arrow b

Kleisli Simulation to Matrix Simulation

$$\begin{tikzcd}
FX & & FY \arrow[ll, "Ff"', "\shortmid" marking] \\
& \sqsubseteq & \\
X \arrow[uu, "c"', "\shortmid" marking] & & Y \arrow[ll, "f" description, "\shortmid" marking] \arrow[uu, "d"', "\shortmid" marking] \\
& \sqsubseteq & \\
& \{\bullet\} \arrow[uul, bend left, "s", "\times" marking] \arrow[uur, bend right, "t"', "\times" marking] &
\end{tikzcd}$$

Forward Kleisli arrow f

$$\begin{tikzcd}
FX \arrow[rr, "Fb", "\shortmid" marking] & & FY \\
& \sqsubseteq & \\
X \arrow[uu, "c"', "\shortmid" marking] \arrow[rr, "b" description, "\shortmid" marking] & & Y \arrow[uu, "d"', "\shortmid" marking] \\
& \sqsubseteq & \\
& \{\bullet\} \arrow[uul, bend left, "s", "\times" marking] \arrow[uur, bend right, "t"', "\times" marking] &
\end{tikzcd}$$

Backward Kleisli arrow b

$$\begin{aligned}
\alpha_{\mathcal{A}} &\sqsubseteq \alpha_{\mathcal{B}} X \\
X \cdot M_{\mathcal{A}}(a) &\sqsubseteq M_{\mathcal{B}}(a) \cdot X \quad (\forall a \in \Sigma) \\
X \beta_{\mathcal{A}} &\sqsubseteq \beta_{\mathcal{B}}
\end{aligned}$$

Forward simulation matrix X

$$\begin{aligned}
\alpha_{\mathcal{A}} X &\sqsubseteq \alpha_{\mathcal{B}} \\
M_{\mathcal{A}}(a) \cdot X &\sqsubseteq X \cdot M_{\mathcal{B}}(a) \quad (\forall a \in \Sigma) \\
\beta_{\mathcal{A}} &\sqsubseteq X \beta_{\mathcal{B}}
\end{aligned}$$

Backward simulation matrix X

Summary of Matrix Simulation

- $\mathcal{S}$**-weighted automaton** is
  automaton whose transitions are weighted with values in $\mathcal{S}$

- **Matrix simulation** between two weighted automata is
  matrix that satisfies some inequalities

$$\begin{aligned} \alpha_{\mathcal{A}} &\sqsubseteq \alpha_{\mathcal{B}} X \\ X \cdot M_{\mathcal{A}}(a) &\sqsubseteq M_{\mathcal{B}}(a) \cdot X \quad (\forall a \in \Sigma) \\ X \beta_{\mathcal{A}} &\sqsubseteq \beta_{\mathcal{B}} \end{aligned}$$

$$\begin{aligned} \alpha_{\mathcal{A}} X &\sqsubseteq \alpha_{\mathcal{B}} \\ M_{\mathcal{A}}(a) \cdot X &\sqsubseteq X \cdot M_{\mathcal{B}}(a) \quad (\forall a \in \Sigma) \\ \beta_{\mathcal{A}} &\sqsubseteq X \beta_{\mathcal{B}} \end{aligned}$$

  - Soundness :
    Existence of simulation matrix implies language inclusion

- Matrix simulation is specialization of **Kleisli simulation**,
  which uses coalgebraic theory

Overview

1. Matrix Simulation
    - Motivation
    - Semiring-Weighted Automaton and Matrix Simulation
    - Origin: from Theory of Coalgebra
2. Partial Execution (to be More "Complete")
3. Specific Examples
    - Example 1 : $\boldsymbol{\mathcal{S}}_{+,\times}$ -weighted Automaton
    - Example 2 : $\boldsymbol{\mathcal{S}}_{\mathbf{max},+}$ -weighted Automaton
4. Conclusion and Future Works

Completeness?

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

or

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

$\mathrm{Lang}(\mathcal{A}) \sqsubseteq \mathrm{Lang}(\mathcal{B})$

Completeness?

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

or

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

$$\xRightarrow{\textbf{soundness}}$$

$$\mathrm{Lang}(\mathcal{A}) \sqsubseteq \mathrm{Lang}(\mathcal{B})$$

Completeness?

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

or

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

$$\xRightarrow{\textbf{soundness}}$$

$$\xcancel{\xLeftarrow[\textbf{completeness}]{}}$$

$\mathrm{Lang}(\mathcal{A}) \sqsubseteq \mathrm{Lang}(\mathcal{B})$

Completeness?

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

or

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

$$\overset{\textbf{soundness}}{\Longrightarrow} \qquad \mathrm{Lang}(\boldsymbol{\mathcal{A}}) \sqsubseteq \mathrm{Lang}(\boldsymbol{\mathcal{B}})$$

$$\underset{\textbf{completeness}}{\xcancel{\Longleftarrow}}$$

$\boxed{\mathrm{Lang}(\boldsymbol{\mathcal{A}}) \sqsubseteq \mathrm{Lang}(\boldsymbol{\mathcal{B}}) \text{ but } \boldsymbol{\mathcal{A}} \not\sqsubseteq_{\mathbf{F}} \boldsymbol{\mathcal{B}},\ \boldsymbol{\mathcal{A}} \not\sqsubseteq_{\mathbf{B}} \boldsymbol{\mathcal{B}}}$

Completeness?

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

or

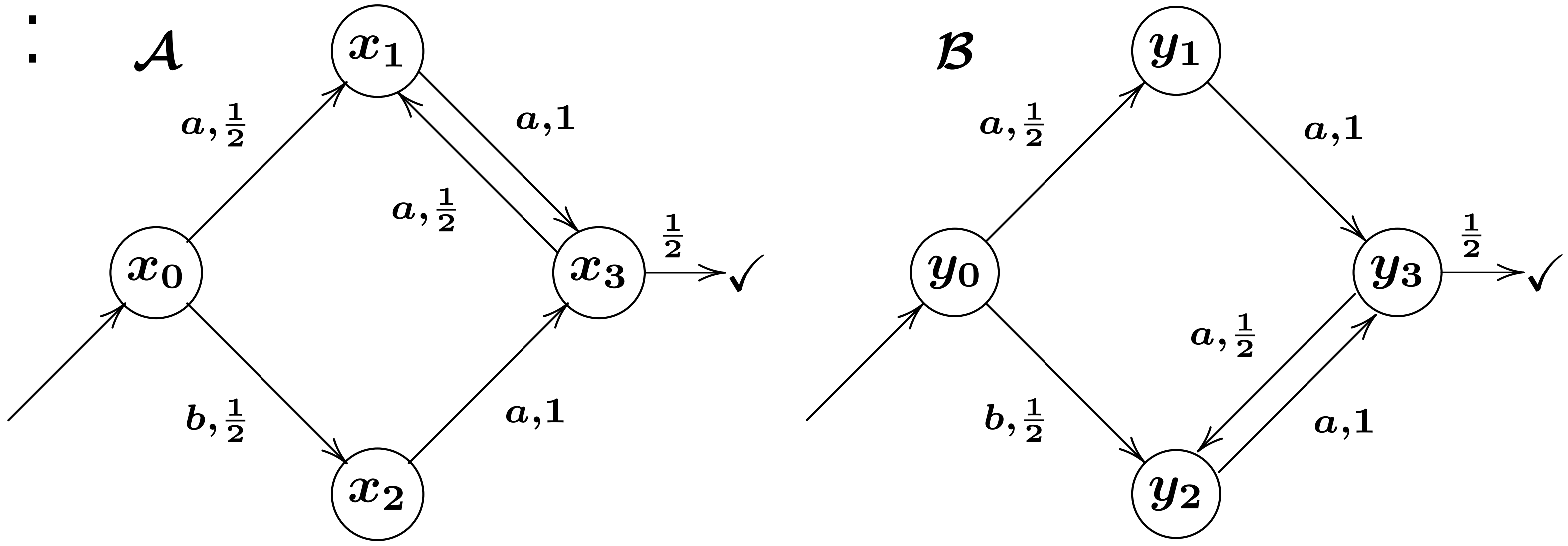
$\mathcal{A} \sqsubseteq_{\mathbf{B}} \mathcal{B}$

$\xRightarrow{\textbf{soundness}}$ $\mathrm{Lang}(\mathcal{A}) \sqsubseteq \mathrm{Lang}(\mathcal{B})$

$\cancel{\xLeftarrow[\textbf{completeness}]{}}$

$\mathrm{Lang}(\mathcal{A}) \sqsubseteq \mathrm{Lang}(\mathcal{B})$ but $\mathcal{A} \not\sqsubseteq_{\mathbf{F}} \mathcal{B}$, $\mathcal{A} \not\sqsubseteq_{\mathbf{B}} \mathcal{B}$

- Example :



Completeness?

$$\begin{array}{c} \mathcal{A} \sqsubseteq_{\mathbf{F}} \mathcal{B} \\ \text{or} \\ \mathcal{A} \sqsubseteq_{\mathbf{B}} \mathcal{B} \end{array} \quad \begin{array}{c} \xRightarrow{\textbf{soundness}} \\ \\ \cancel{\xLeftarrow[\textbf{completeness}]{}} \end{array} \quad \mathrm{Lang}(\mathcal{A}) \sqsubseteq \mathrm{Lang}(\mathcal{B})$$

- Language inclusion is **undecidable** for

$$\begin{cases} \mathcal{S}_{+,\times}\text{-weighted automata [Blondel \& Canterni, 2003]} \\ \mathcal{S}_{\mathbf{max},+}\text{-weighted automata [Krob, 1992]} \end{cases}$$

- Existence of Fwd. / Bwd. matrix simulation is **decidable** for them [Tarski, 1951]

Partial Execution

- Transformation of weighted automata

Partial Execution

- Transformation of weighted automata

- Two types : {
  - Forward Partial Execution (**FPE**)
  - Backward Partial Execution (**BPE**)

Partial Execution

- Transformation of weighted automata
- Two types :
  - Forward Partial Execution (**FPE**)
  - Backward Partial Execution (**BPE**)
- Produce matrix simulation

Partial Execution

- Transformation of weighted automata
- Two types : $\left\{\begin{array}{l}\text{Forward Partial Execution (FPE)} \\ \text{Backward Partial Execution (BPE)}\end{array}\right.$
- Produce matrix simulation

i.e. It can be that

$$\mathrm{Lang}(\boldsymbol{\mathcal{A}}) \sqsubseteq \mathrm{Lang}(\boldsymbol{\mathcal{B}})$$

but $\boldsymbol{\mathcal{A}} \not\sqsubseteq_{\mathbf{F}} \boldsymbol{\mathcal{B}}$

$$\mathrm{Lang}(\boldsymbol{\mathcal{A}}) \sqsubseteq \mathrm{Lang}(\boldsymbol{\mathcal{B}})$$

but $\boldsymbol{\mathcal{A}} \not\sqsubseteq_{\mathbf{B}} \boldsymbol{\mathcal{B}}$

Partial Execution

- Transformation of weighted automata
- Two types : { Forward Partial Execution (**FPE**)
  Backward Partial Execution (**BPE**)
- Produce matrix simulation

i.e. It can be that

$$\mathrm{Lang}(\mathcal{A}) \sqsubseteq \mathrm{Lang}(\mathcal{B})$$

but $\mathcal{A} \not\sqsubseteq_{\mathbf{F}} \mathcal{B}$

$$\mathrm{FPE}(\mathcal{A}) \sqsubseteq_{\mathbf{F}} \mathrm{BPE}(\mathcal{B})$$

where

$$\mathrm{Lang}(\mathcal{A}) = \mathrm{Lang}(\mathrm{FPE}(\mathcal{A}))$$
$$\mathrm{Lang}(\mathcal{B}) = \mathrm{Lang}(\mathrm{BPE}(\mathcal{B}))$$

$$\mathrm{Lang}(\mathcal{A}) \sqsubseteq \mathrm{Lang}(\mathcal{B})$$

but $\mathcal{A} \not\sqsubseteq_{\mathrm{B}} \mathcal{B}$

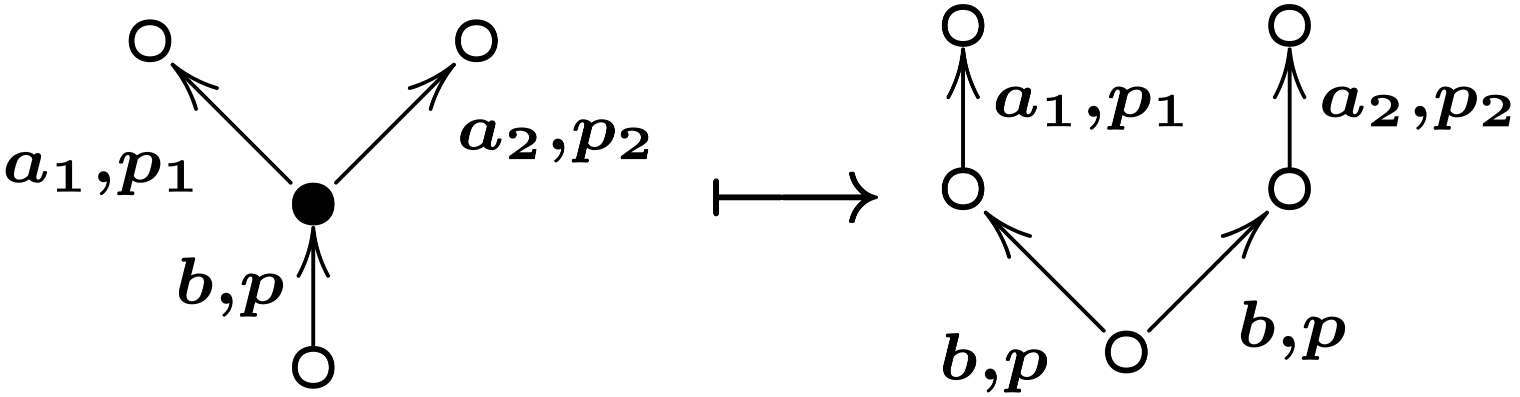
$$\mathrm{BPE}(\mathcal{A}) \sqsubseteq_{\mathrm{B}} \mathrm{FPE}(\mathcal{B})$$

where

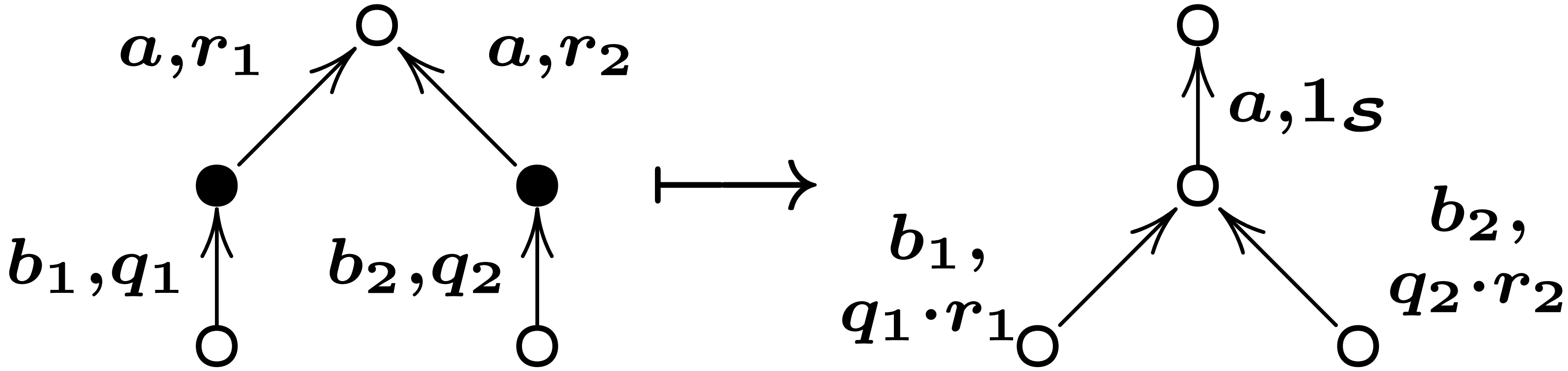
$$\mathrm{Lang}(\mathcal{A}) = \mathrm{Lang}(\mathrm{BPE}(\mathcal{A}))$$
$$\mathrm{Lang}(\mathcal{B}) = \mathrm{Lang}(\mathrm{FPE}(\mathcal{B}))$$

Forward Partial Execution

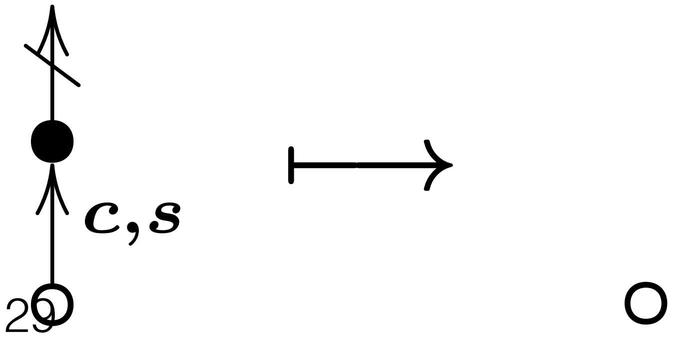
- Pictorially,
  - Split backward



  - Merge backward

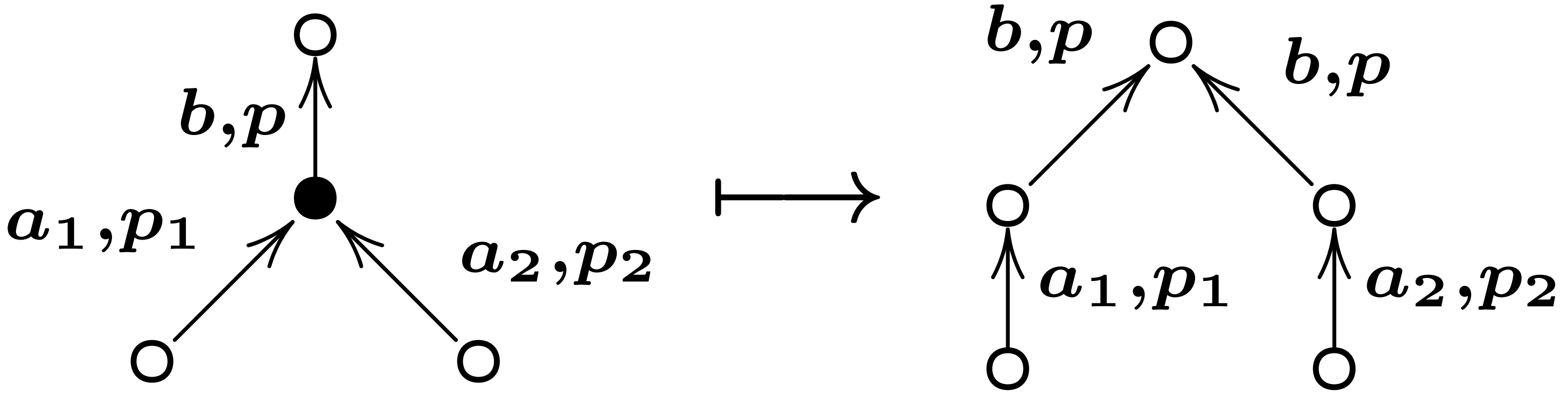


  - Eliminate dead end

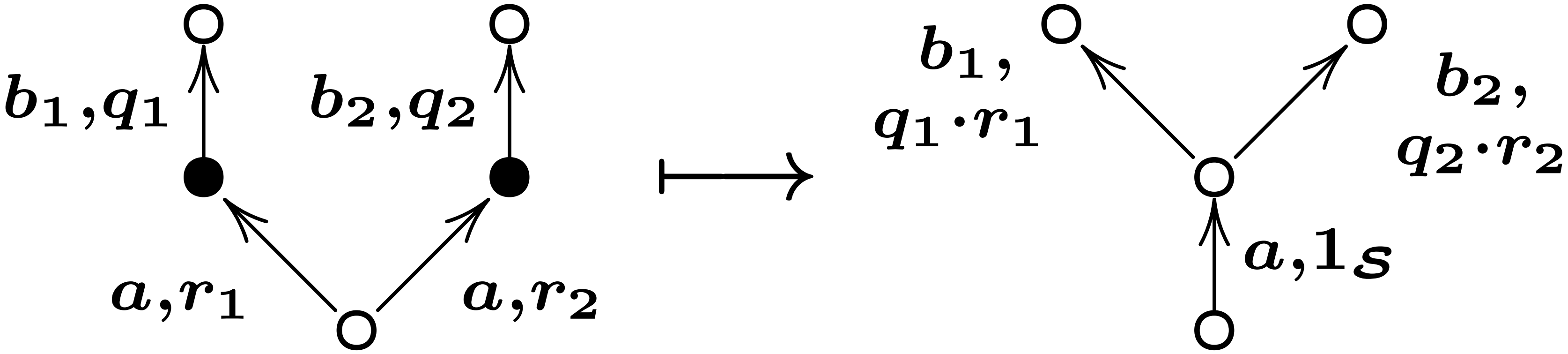


Backward Partial Execution

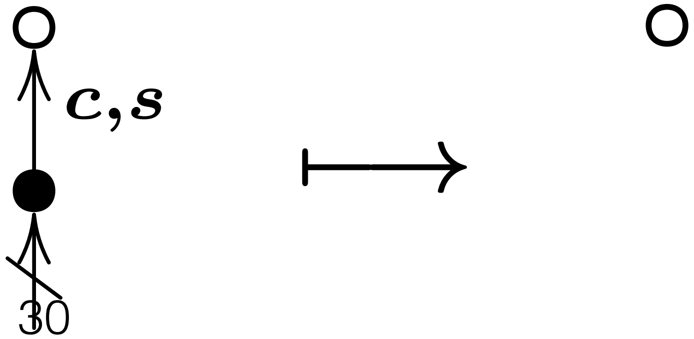
- Pictorially,
  - Split forward



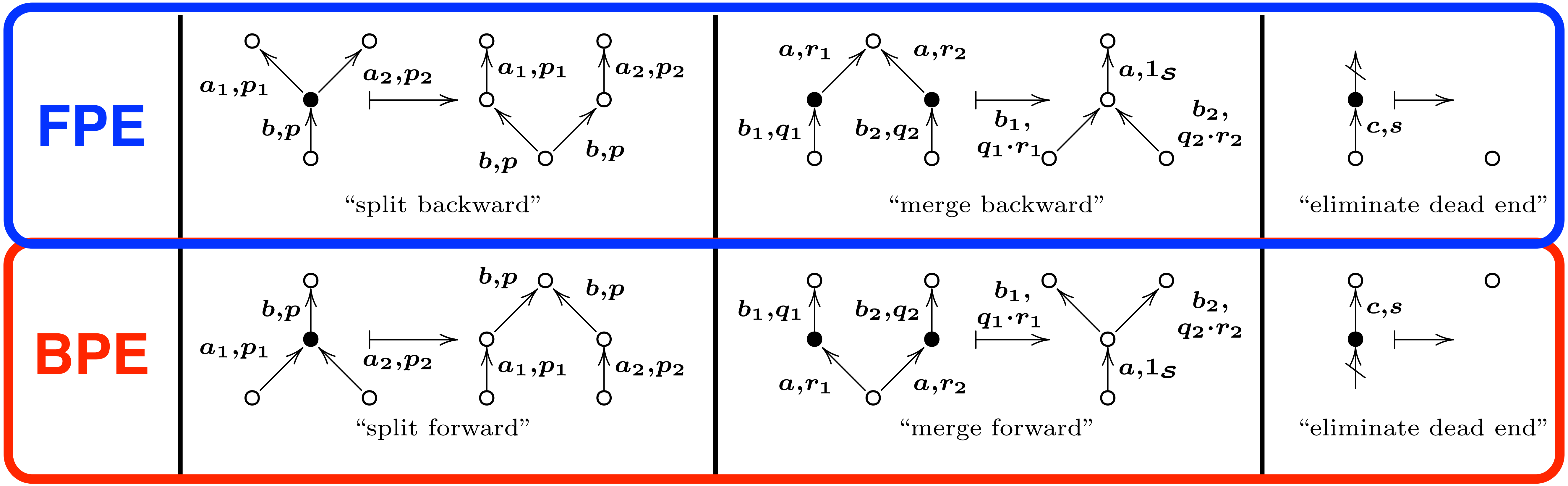
  - Merge forward



  - Eliminate dead end



Usage of Execution



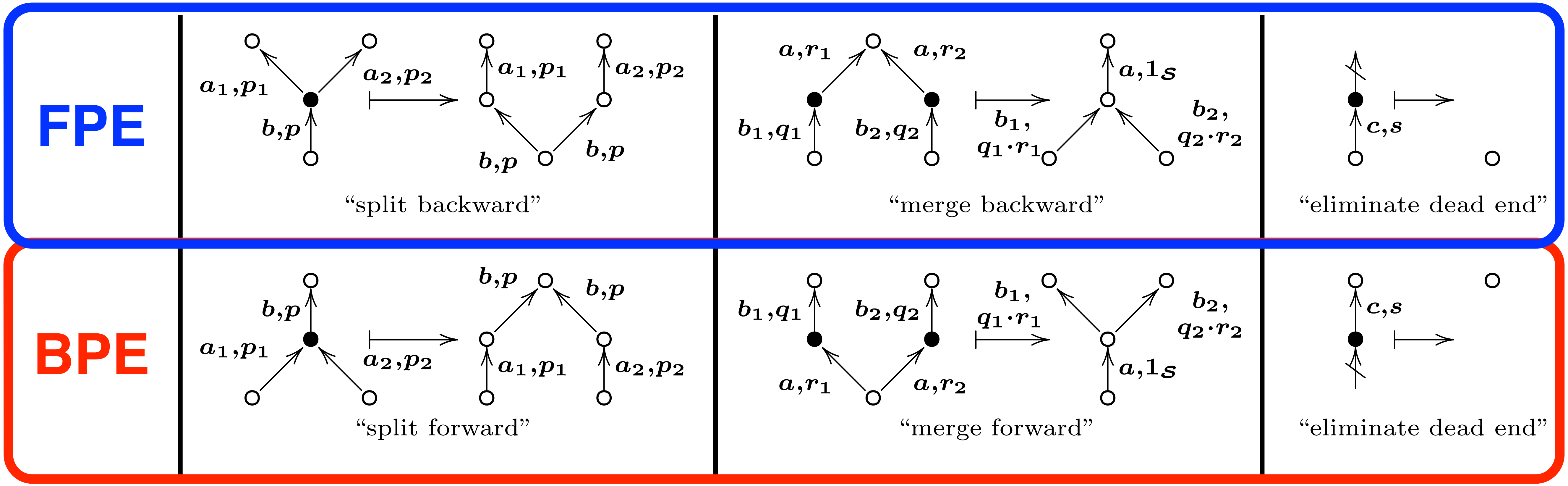
“split backward” “merge backward” “eliminate dead end”

“split forward” “merge forward” “eliminate dead end”

- **FPE** and **BPE** can increase matrix simulation only if applied to proper side of proper simulation

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

Usage of Execution



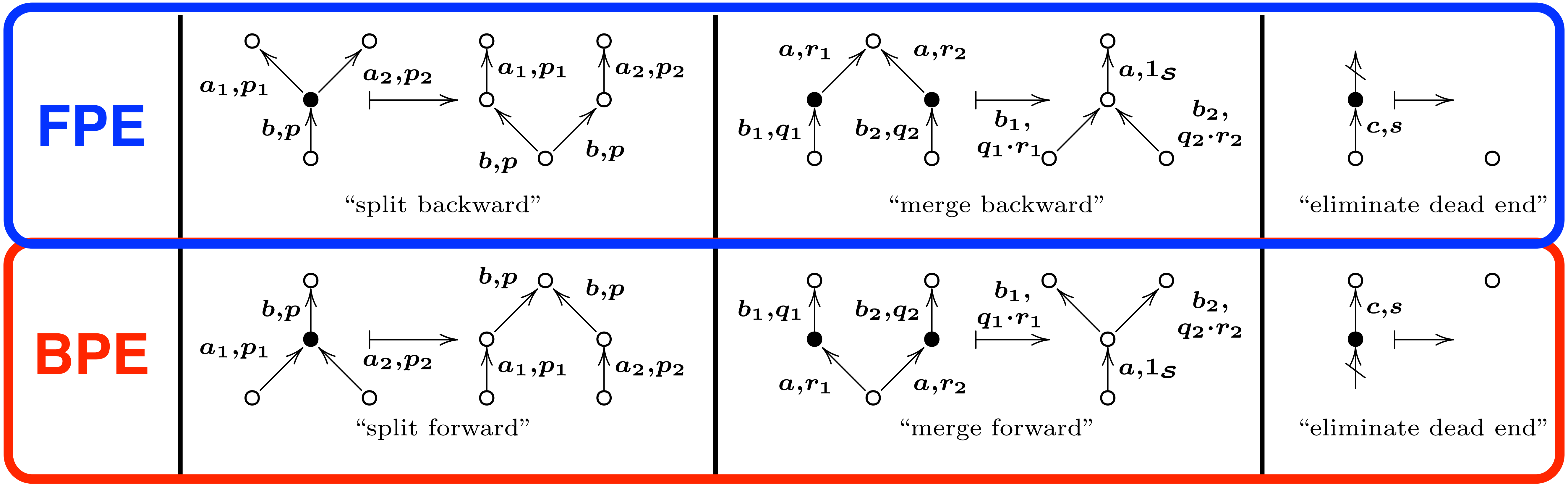
“split backward” “merge backward” “eliminate dead end”

“split forward” “merge forward” “eliminate dead end”

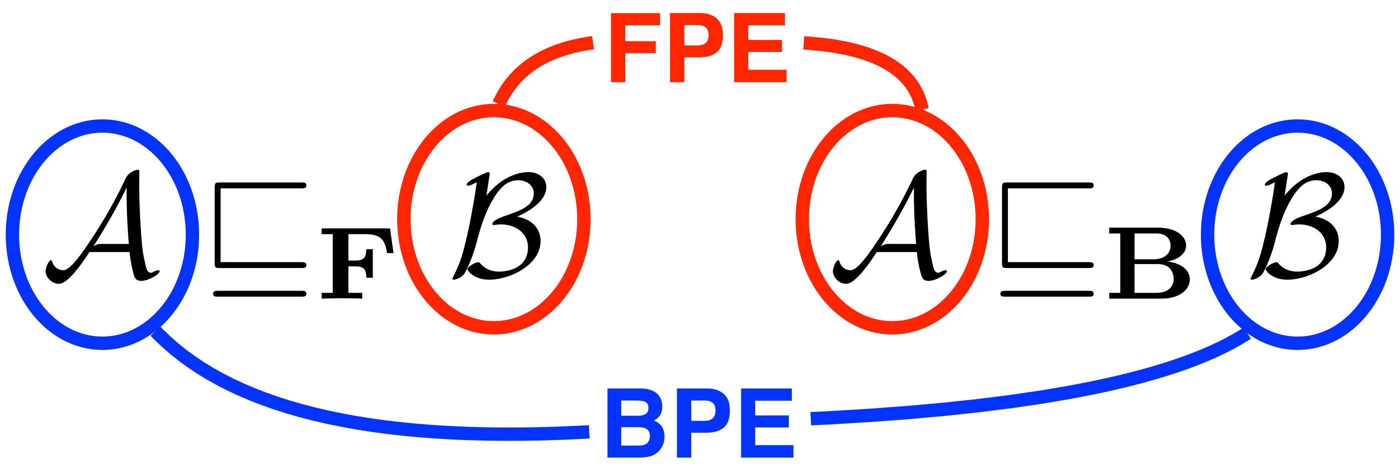
- **FPE** and **BPE** can increase matrix simulation only if applied to proper side of proper simulation

$$\mathcal{A} \sqsubseteq_{\mathbf{F}} \overset{\text{FPE}}{\boxed{\mathcal{B}}} \qquad \overset{\text{FPE}}{\boxed{\mathcal{A}}} \sqsubseteq_{\mathbf{B}} \mathcal{B}$$

Usage of Execution



- **FPE** and **BPE** can increase matrix simulation only if applied to <u>proper side</u> of <u>proper simulation</u>



Soundness and Adequacy

- **Soundness**

$$\mathrm{Lang}(\mathcal{A}) \sqsubseteq \mathrm{Lang}(\mathcal{B})$$

- **Adequacy**

Soundness and Adequacy

- **Soundness**

$$\begin{tikzcd}
\mathrm{FPE}(\boldsymbol{\mathcal{A}}) \sqsubseteq_{\mathbf{F}} \mathrm{BPE}(\boldsymbol{\mathcal{B}}) \arrow[dr, Rightarrow] & \\
& \mathrm{Lang}(\boldsymbol{\mathcal{A}}) \sqsubseteq \mathrm{Lang}(\boldsymbol{\mathcal{B}}) \\
\mathrm{BPE}(\boldsymbol{\mathcal{A}}) \sqsubseteq_{\mathbf{B}} \mathrm{FPE}(\boldsymbol{\mathcal{B}}) \arrow[ur, Rightarrow] &
\end{tikzcd}$$

➡ Properly applied transformation maintains soundness

- **Adequacy**

Soundness and Adequacy

- **Soundness**

$$\begin{tikzcd}
\mathrm{FPE}(\mathcal{A}) \sqsubseteq_{\mathbf{F}} \mathrm{BPE}(\mathcal{B}) \arrow[dr, Rightarrow] & \\
& \mathrm{Lang}(\mathcal{A}) \sqsubseteq \mathrm{Lang}(\mathcal{B}) \\
\mathrm{BPE}(\mathcal{A}) \sqsubseteq_{\mathbf{B}} \mathrm{FPE}(\mathcal{B}) \arrow[ur, Rightarrow] &
\end{tikzcd}$$

➡ Properly applied transformation maintains soundness

- **Adequacy**

$$\mathcal{A} \sqsubseteq_{\mathbf{F}} \mathcal{B} \quad \Longrightarrow \quad \mathrm{FPE}(\mathcal{A}) \sqsubseteq_{\mathbf{F}} \mathrm{BPE}(\mathcal{B})$$

$$\mathcal{A} \sqsubseteq_{\mathbf{B}} \mathcal{B} \quad \Longrightarrow \quad \mathrm{BPE}(\mathcal{A}) \sqsubseteq_{\mathbf{B}} \mathrm{FPE}(\mathcal{B})$$

➡ Properly applied transformation does not destroy simulation

Coalgebraic Characterization of Partial Execution

Matrix simulation

$\mathcal{S}$ -weighted automaton

$$\begin{aligned} \alpha_{\mathcal{A}} &\sqsubseteq \alpha_{\mathcal{B}} X \ , \\ X \cdot M_{\mathcal{A}}(a) &\sqsubseteq M_{\mathcal{B}}(a) \cdot X \\ &\qquad (\forall a \in \Sigma) \ , \\ X \beta_{\mathcal{A}} &\sqsubseteq \beta_{\mathcal{B}} \end{aligned}$$

Coalgebraic Characterization of Partial Execution

FPE
BPE

Matr

$\mathcal{S}$ -weighted automaton

$$
\begin{aligned}
\alpha_{\mathcal{A}} &\sqsubseteq \alpha_{\mathcal{B}} X \quad , \\
X \cdot M_{\mathcal{A}}(a) &\sqsubseteq M_{\mathcal{B}}(a) \cdot X \\
&\qquad (\forall a \in \Sigma) \quad , \\
X \beta_{\mathcal{A}} &\sqsubseteq \beta_{\mathcal{B}}
\end{aligned}
$$

Coalgebraic Characterization of Partial Execution

Kleisli simulation by Hasuo (2006)

Systems represented as coalgebra

$$\begin{tikzcd}
FX & & FY \arrow[ll, "Ff"', "|" marking] \\
X \arrow[u, "c", "|" marking] \arrow[urr, phantom, "\sqsubseteq"] & & Y \arrow[ll, "f", "|" marking] \arrow[u, "d"', "|" marking] \\
& \{\bullet\} \arrow[ul, bend left, "s", "|" marking] \arrow[ur, bend right, "t"', "|" marking] \arrow[u, phantom, "\sqsubseteq"] &
\end{tikzcd}$$

specialize

FPE

BPE

Matr

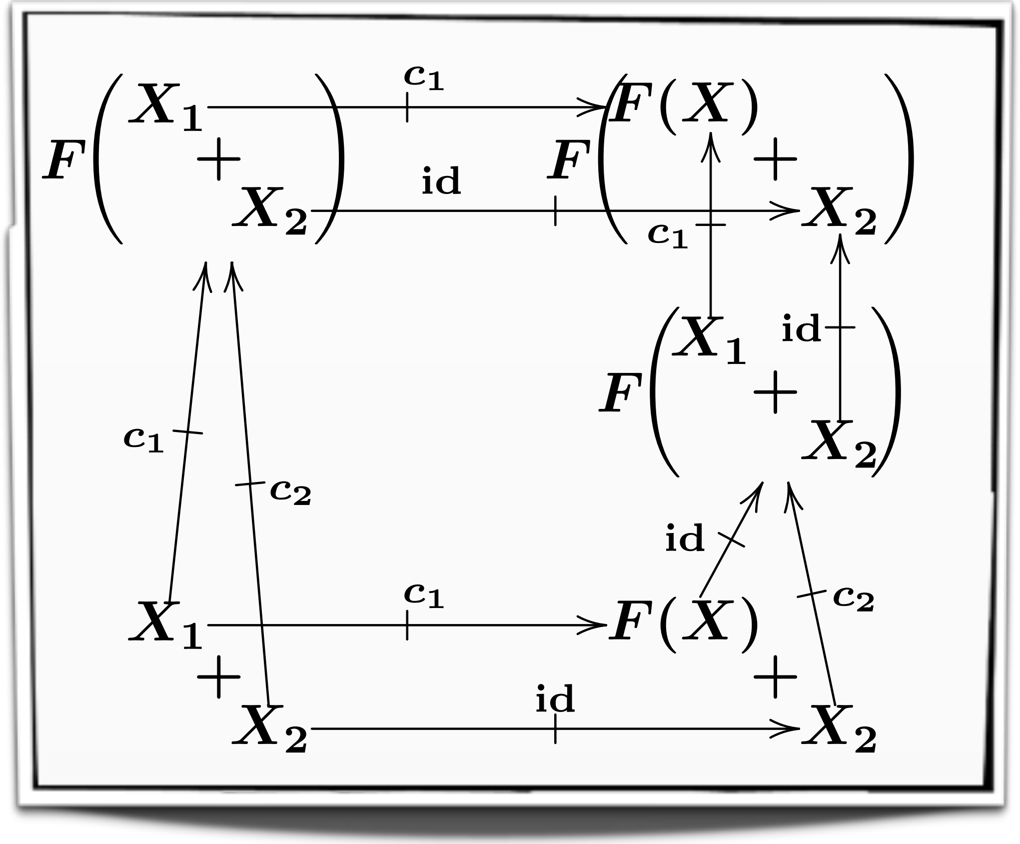
$\mathcal{S}$ -weighted automaton

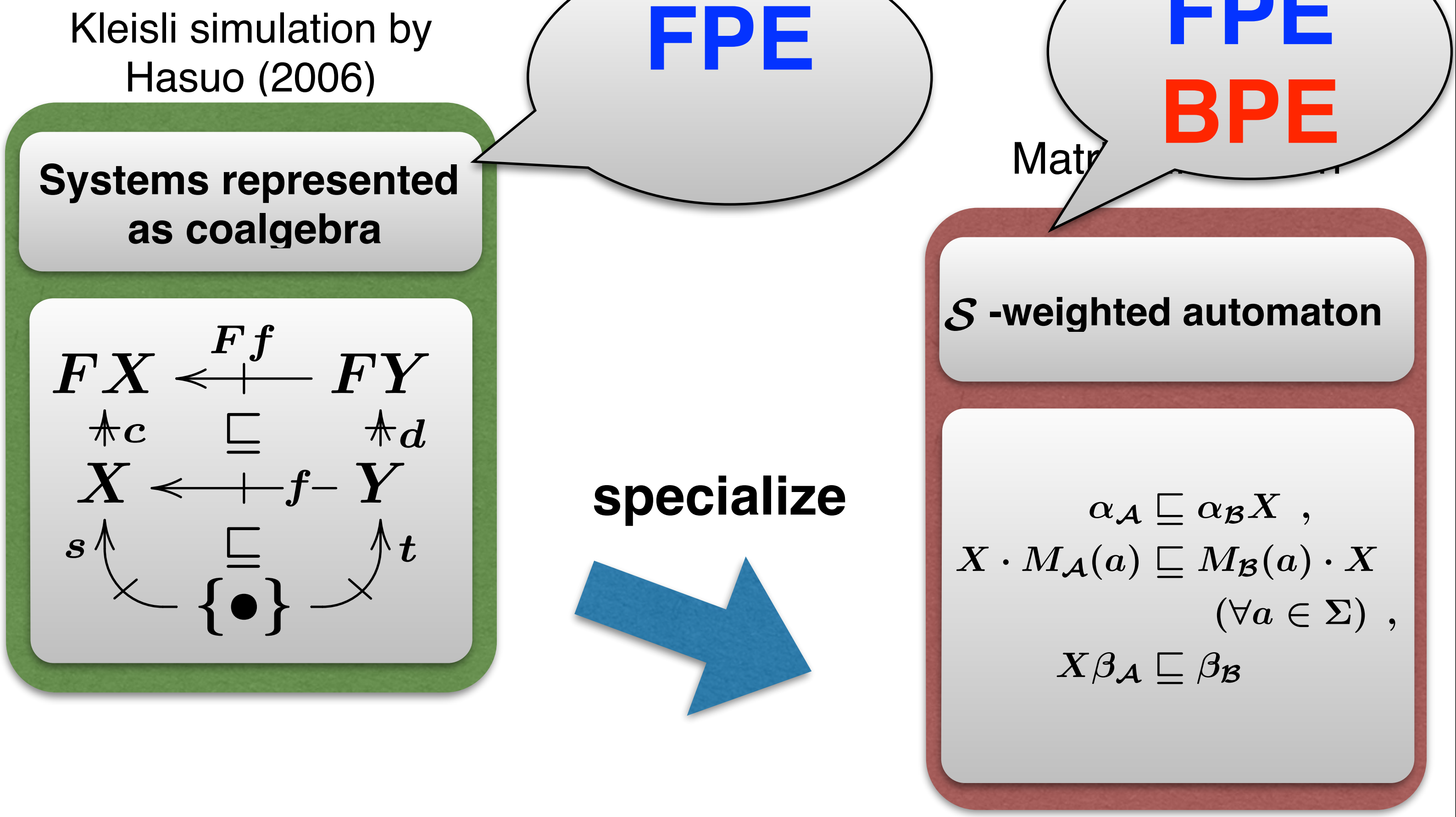
$$\alpha_{\mathcal{A}} \sqsubseteq \alpha_{\mathcal{B}} X \ ,$$

$$X \cdot M_{\mathcal{A}}(a) \sqsubseteq M_{\mathcal{B}}(a) \cdot X \quad (\forall a \in \Sigma) \ ,$$

$$X \beta_{\mathcal{A}} \sqsubseteq \beta_{\mathcal{B}}$$

Coalgebraic Characterization of Partial Execution





Kleisli simulation by Hasuo (2006)

FPE

Systems represented as coalgebra

$$\begin{tikzcd}
FX & & FY \arrow[ll, "Ff"'] \\
X \arrow[u, "c"] \arrow[urr, phantom, "\sqsubseteq"] & & Y \arrow[u, "d"'] \arrow[ll, "f"] \\
& \{\bullet\} \arrow[ul, bend left, "s"] \arrow[ur, bend right, "t"'] \arrow[u, phantom, "\sqsubseteq"] &
\end{tikzcd}$$

specialize

Matr

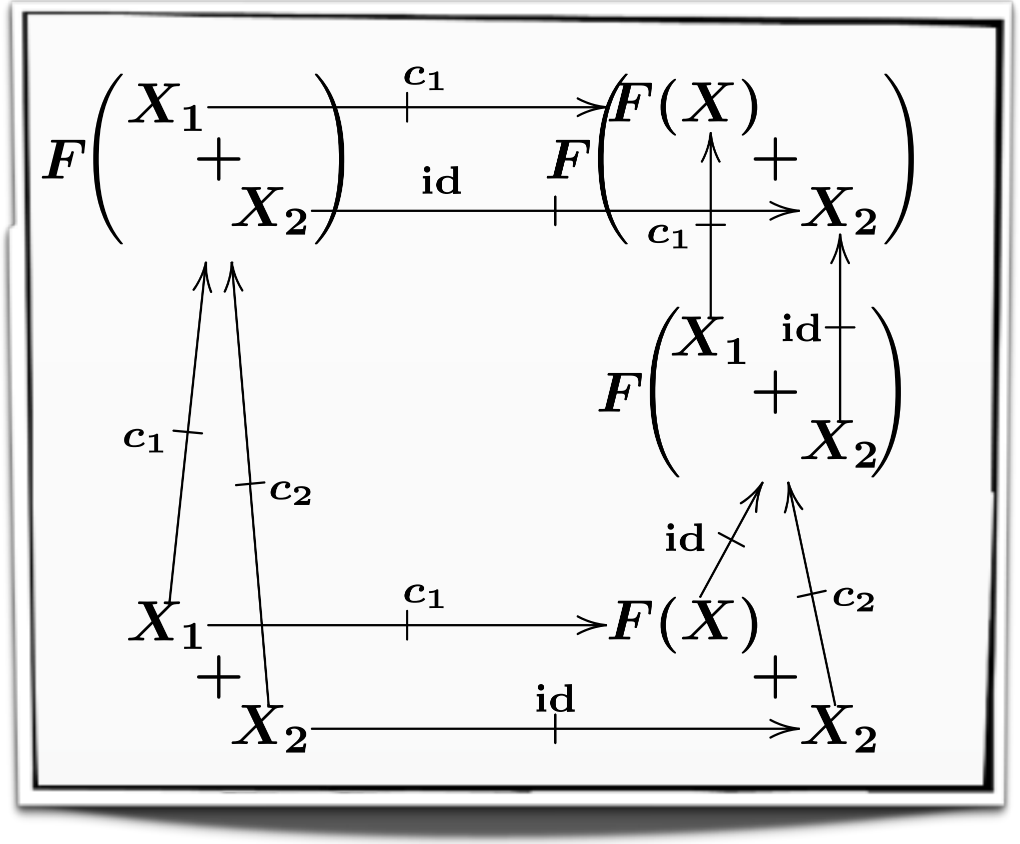
FPE
BPE

$\mathcal{S}$ -weighted automaton

$$\alpha_{\mathcal{A}} \sqsubseteq \alpha_{\mathcal{B}} X \ ,$$
$$X \cdot M_{\mathcal{A}}(a) \sqsubseteq M_{\mathcal{B}}(a) \cdot X \quad (\forall a \in \Sigma) \ ,$$
$$X \beta_{\mathcal{A}} \sqsubseteq \beta_{\mathcal{B}}$$

- We can define **fwd.** partial execution for **Kleisli simulation**

Coalgebraic Characterization of Partial Execution



Kleisli simulation by Hasuo (2006)

Systems represented as coalgebra

$$\begin{tikzcd}
F X & & F Y \arrow[ll, "Ff"'] \\
 & \sqsubseteq & \\
X \arrow[uu, "c"] & & Y \arrow[ll, "f"'] \arrow[uu, "d"'] \\
 & \sqsubseteq & \\
 & \{\bullet\} \arrow[uul, "s"] \arrow[uur, "t"'] &
\end{tikzcd}$$

FPE

specialize

Matr

FPE
BPE

$\mathcal{S}$ -weighted automaton

$$\alpha_{\mathcal{A}} \sqsubseteq \alpha_{\mathcal{B}} X \;,$$
$$X \cdot M_{\mathcal{A}}(a) \sqsubseteq M_{\mathcal{B}}(a) \cdot X \quad (\forall a \in \Sigma) \;,$$
$$X \beta_{\mathcal{A}} \sqsubseteq \beta_{\mathcal{B}}$$

- We can define **fwd**. partial execution for **Kleisli simulation**
- How about **bwd**. partial execution?
    - ➡ “Opposite automaton” should be defined?

Overview

1. Matrix Simulation
    - Motivation
    - Semiring-Weighted Automaton and Matrix Simulation
    - Origin: from Theory of Coalgebra
2. Partial Execution (to be More “Complete”)
3. Specific Examples
    - Example 1 : $\boldsymbol{\mathcal{S}}_{+,\times}$-weighted Automaton
    - Example 2 : $\boldsymbol{\mathcal{S}}_{\mathbf{max},+}$ -weighted Automaton
4. Conclusion and Future Works

Comparison with Other Simulations for $\mathcal{S}_{+,\times}$-Weighted Automata

$$\mathcal{S}_{+,\times} = ([\mathbf{0}, \infty], +, \mathbf{0}, \times, \mathbf{1}, \leq)$$

Model for probabilistic system

Comparison with Other Simulations for $\mathcal{S}_{+,\times}$ -Weighted Automata

$$\mathcal{S}_{+,\times} = ([0, \infty], +, 0, \times, 1, \leq)$$

Model for probabilistic system

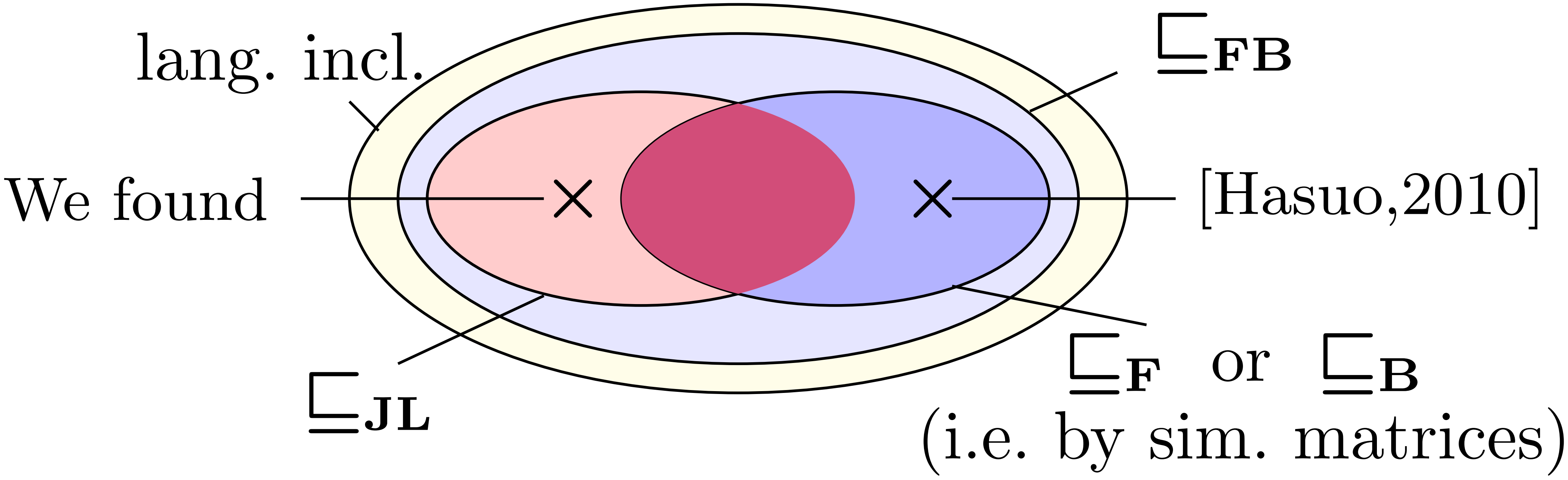
- Simulation by Jonsson & Larsen (1991) (JL-simulation)

Comparison with Other Simulations for $\mathcal{S}_{+,\times}$-Weighted Automata

$$\mathcal{S}_{+,\times} = ([\mathbf{0}, \infty], +, \mathbf{0}, \times, \mathbf{1}, \leq)$$

Model for
probabilistic system

- Simulation by Jonsson & Larsen (1991) (JL-simulation)



Experimental Results 1

- Linear inequalities for matrix simulation are “ordinal” linear inequalities

→ We implemented using linear programming solver

Experimental Results 1

- Linear inequalities for matrix simulation are “ordinal” linear inequalities
  → We implemented using linear programming solver
- Verification of anonymity of Grade protocol [Kiefer et al. 2011]

Experimental Results 1

- Linear inequalities for matrix simulation are “ordinal” linear inequalities

  ➡ We implemented using linear programming solver

- Verification of anonymity of Grade protocol [Kiefer et al. 2011]

- **In** [Kiefer et al. 2011] ,

Experimental Results 1

- Linear inequalities for matrix simulation are “ordinal” linear inequalities

  ➡ We implemented using linear programming solver

- Verification of anonymity of Grade protocol [Kiefer et al. 2011]

- In [Kiefer et al. 2011],
  - Programs **P** and **S** are introduced
    - **Equivalence** of **P** and **S** implies anonymity of **P**

Program of Grade protocol

Obviously satisfies anonymity

Experimental Results 1

- Linear inequalities for matrix simulation are “ordinal” linear inequalities
  ➡ We implemented using linear programming solver
- Verification of anonymity of Grade protocol [Kiefer et al. 2011]

Program of Grade protocol

Obviously satisfies anonymity

- In [Kiefer et al. 2011],
  - Programs **P** and **S** are introduced
    - **Equivalence** of **P** and **S** implies anonymity of **P**
  - APEX (tool that change program to $\mathcal{S}_{+,\times}$-weighted automaton) is implemented

$$\mathbf{P} \mapsto \mathcal{A}_{\mathbf{P}} \qquad \mathbf{S} \mapsto \mathcal{A}_{\mathbf{S}}$$

Experimental Results 1

- Linear inequalities for matrix simulation are “ordinal” linear inequalities

  ➡ We implemented using linear programming solver

- Verification of anonymity of Grade protocol [Kiefer et al. 2011]

- In [Kiefer et al. 2011],

  Program of Grade protocol

  Obviously satisfies anonymity

  - Programs **P** and **S** are introduced
    - **Equivalence** of **P** and **S** implies anonymity of **P**
  - APEX (tool that change program to $\mathcal{S}_{+,\times}$-weighted automaton) is implemented

    $$\mathbf{P} \mapsto \mathcal{A}_{\mathbf{P}} \qquad \mathbf{S} \mapsto \mathcal{A}_{\mathbf{S}}$$

  - Experiment on equivalence check of produced $\mathcal{S}_{+,\times}$-weighted automata

    $$\mathrm{Lang}(\mathcal{A}_{\mathbf{P}}) = \mathrm{Lang}(\mathcal{A}_{\mathbf{S}})$$

Experimental Results 1

- Linear inequalities for matrix simulation are “ordinal” linear inequalities

  → We implemented using linear programming solver

- Verification of anonymity of Grade protocol [Kiefer et al. 2011]

Program of Grade protocol

Obviously satisfies anonymity

- In [Kiefer et al. 2011],
  - Programs **P** and **S** are introduced
    - **Equivalence** of **P** and **S** implies anonymity of **P**
  - APEX (tool that change program to $\mathcal{S}_{+,\times}$-weighted automaton) is implemented

$$\mathbf{P} \mapsto \mathcal{A}_{\mathbf{P}} \qquad \mathbf{S} \mapsto \mathcal{A}_{\mathbf{S}}$$

  - Experiment on equivalence check of produced $\mathcal{S}_{+,\times}$-weighted automata

$$\mathrm{Lang}(\mathcal{A}_{\mathbf{P}}) = \mathrm{Lang}(\mathcal{A}_{\mathbf{S}})$$

- We proved this equivalence by two-way language inclusion

$$\mathrm{Lang}(\mathcal{A}_{\mathbf{P}}) = \mathrm{Lang}(\mathcal{A}_{\mathbf{S}}) \Longleftarrow \begin{cases} \mathrm{Lang}(\mathcal{A}_{\mathbf{P}}) \sqsubseteq \mathrm{Lang}(\mathcal{A}_{\mathbf{S}}) \text{ and} \\ \mathrm{Lang}(\mathcal{A}_{\mathbf{P}}) \sqsupseteq \mathrm{Lang}(\mathcal{A}_{\mathbf{S}}) \end{cases}$$

Experimental Results 1

param.		$\mathcal{A}_\mathsf{P}$		$\mathcal{A}_\mathsf{S}$			direction,	time	space
$\boldsymbol{G}$	$\boldsymbol{S}$	#st.	#tr.	#st.	#tr.	$\vert\boldsymbol{\Sigma}\vert$	fwd./bwd.	(sec)	(GB)
2	8	578	1522	130	642	11	$\mathcal{A}_\mathsf{P} \sqsubseteq_\mathbf{F} \mathcal{A}_\mathsf{S}$	1.77	1.21
							$\mathcal{A}_\mathsf{P} \sqsupseteq_\mathbf{B} \mathcal{A}_\mathsf{S}$	1.72	1.22
2	10	1102	2982	202	1202	13	$\mathcal{A}_\mathsf{P} \sqsubseteq_\mathbf{F} \mathcal{A}_\mathsf{S}$	9.42	4.05
							$\mathcal{A}_\mathsf{P} \sqsupseteq_\mathbf{B} \mathcal{A}_\mathsf{S}$	9.25	4.09
2	12	1874	5162	290	2018	15	$\mathcal{A}_\mathsf{P} \sqsubseteq_\mathbf{F} \mathcal{A}_\mathsf{S}$	38.60	11.51
							$\mathcal{A}_\mathsf{P} \sqsupseteq_\mathbf{B} \mathcal{A}_\mathsf{S}$	38.34	11.63
3	8	1923	7107	243	2163	20	$\mathcal{A}_\mathsf{P} \sqsubseteq_\mathbf{F} \mathcal{A}_\mathsf{S}$	44.43	12.26
							$\mathcal{A}_\mathsf{P} \sqsupseteq_\mathbf{B} \mathcal{A}_\mathsf{S}$	44.11	12.64
4	6	1636	7468	196	1924	23	$\mathcal{A}_\mathsf{P} \sqsubseteq_\mathbf{F} \mathcal{A}_\mathsf{S}$	30.28	10.39
							$\mathcal{A}_\mathsf{P} \sqsupseteq_\mathbf{B} \mathcal{A}_\mathsf{S}$	29.94	10.49

Experimental Results 1

param.		$\mathcal{A}_\mathsf{P}$		$\mathcal{A}_\mathsf{S}$			direction,	time	space
G	S	#st.	#tr.	#st.	#tr.	$\|\boldsymbol{\Sigma}\|$	fwd./bwd.	(sec)	(GB)
2	8	578	1522	130	642	11	$\mathcal{A}_\mathsf{P} \sqsubseteq_\mathbf{F} \mathcal{A}_\mathsf{S}$	1.77	1.21
							$\mathcal{A}_\mathsf{P} \sqsupseteq_\mathbf{B} \mathcal{A}_\mathsf{S}$	1.72	1.22
2	10	1102	2982	202	1202	13	$\mathcal{A}_\mathsf{P} \sqsubseteq_\mathbf{F} \mathcal{A}_\mathsf{S}$	9.42	4.05
							$\mathcal{A}_\mathsf{P} \sqsupseteq_\mathbf{B} \mathcal{A}_\mathsf{S}$	9.25	4.09
2	12	1874	5162	290	2018	15	$\mathcal{A}_\mathsf{P} \sqsubseteq_\mathbf{F} \mathcal{A}_\mathsf{S}$	38.60	11.51
							$\mathcal{A}_\mathsf{P} \sqsupseteq_\mathbf{B} \mathcal{A}_\mathsf{S}$	38.34	11.63
3	8	1923	7107	243	2163	20	$\mathcal{A}_\mathsf{P} \sqsubseteq_\mathbf{F} \mathcal{A}_\mathsf{S}$	44.43	12.26
							$\mathcal{A}_\mathsf{P} \sqsupseteq_\mathbf{B} \mathcal{A}_\mathsf{S}$	44.11	12.64
4	6	1636	7468	196	1924	23	$\mathcal{A}_\mathsf{P} \sqsubseteq_\mathbf{F} \mathcal{A}_\mathsf{S}$	30.28	10.39
							$\mathcal{A}_\mathsf{P} \sqsupseteq_\mathbf{B} \mathcal{A}_\mathsf{S}$	29.94	10.49

- Two-way inclusion could be checked for all parameters
    - $\mathcal{A}_P \sqsubseteq_\mathbf{F} \mathcal{A}_S$ and $\mathcal{A}_S \sqsubseteq_\mathbf{B} \mathcal{A}_P$ were found

Experimental Results 1

param.		$\mathcal{A}_\mathsf{P}$		$\mathcal{A}_\mathsf{S}$			direction,	time	space
G	S	#st.	#tr.	#st.	#tr.	$\lvert\Sigma\rvert$	fwd./bwd.	(sec)	(GB)
2	8	578	1522	130	642	11	$\mathcal{A}_\mathsf{P} \sqsubseteq_\mathbf{F} \mathcal{A}_\mathsf{S}$	1.77	1.21
							$\mathcal{A}_\mathsf{P} \sqsupseteq_\mathbf{B} \mathcal{A}_\mathsf{S}$	1.72	1.22
2	10	1102	2982	202	1202	13	$\mathcal{A}_\mathsf{P} \sqsubseteq_\mathbf{F} \mathcal{A}_\mathsf{S}$	9.42	4.05
							$\mathcal{A}_\mathsf{P} \sqsupseteq_\mathbf{B} \mathcal{A}_\mathsf{S}$	9.25	4.09
2	12	1874	5162	290	2018	15	$\mathcal{A}_\mathsf{P} \sqsubseteq_\mathbf{F} \mathcal{A}_\mathsf{S}$	38.60	11.51
							$\mathcal{A}_\mathsf{P} \sqsupseteq_\mathbf{B} \mathcal{A}_\mathsf{S}$	38.34	11.63
3	8	1923	7107	243	2163	20	$\mathcal{A}_\mathsf{P} \sqsubseteq_\mathbf{F} \mathcal{A}_\mathsf{S}$	44.43	12.26
							$\mathcal{A}_\mathsf{P} \sqsupseteq_\mathbf{B} \mathcal{A}_\mathsf{S}$	44.11	12.64
4	6	1636	7468	196	1924	23	$\mathcal{A}_\mathsf{P} \sqsubseteq_\mathbf{F} \mathcal{A}_\mathsf{S}$	30.28	10.39
							$\mathcal{A}_\mathsf{P} \sqsupseteq_\mathbf{B} \mathcal{A}_\mathsf{S}$	29.94	10.49

- Two-way inclusion could be checked for all parameters
  - $\mathcal{A}_P \sqsubseteq_\mathbf{F} \mathcal{A}_S$ and $\mathcal{A}_S \sqsubseteq_\mathbf{B} \mathcal{A}_P$ were found
- Space is serious problem

Experimental Results 1

param.		$\mathcal{A}_\mathsf{P}$		$\mathcal{A}_\mathsf{S}$			direction,	time	space
G	S	#st.	#tr.	#st.	#tr.	$\lvert\Sigma\rvert$	fwd./bwd.	(sec)	(GB)
2	8	578	1522	130	642	11	$\mathcal{A}_\mathsf{P} \sqsubseteq_\mathbf{F} \mathcal{A}_\mathsf{S}$	1.77	1.21
							$\mathcal{A}_\mathsf{P} \sqsupseteq_\mathbf{B} \mathcal{A}_\mathsf{S}$	1.72	1.22
2	10	1102	2982	202	1202	13	$\mathcal{A}_\mathsf{P} \sqsubseteq_\mathbf{F} \mathcal{A}_\mathsf{S}$	9.42	4.05
							$\mathcal{A}_\mathsf{P} \sqsupseteq_\mathbf{B} \mathcal{A}_\mathsf{S}$	9.25	4.09
2	12	1874	5162	290	2018	15	$\mathcal{A}_\mathsf{P} \sqsubseteq_\mathbf{F} \mathcal{A}_\mathsf{S}$	38.60	11.51
							$\mathcal{A}_\mathsf{P} \sqsupseteq_\mathbf{B} \mathcal{A}_\mathsf{S}$	38.34	11.63
3	8	1923	7107	243	2163	20	$\mathcal{A}_\mathsf{P} \sqsubseteq_\mathbf{F} \mathcal{A}_\mathsf{S}$	44.43	12.26
							$\mathcal{A}_\mathsf{P} \sqsupseteq_\mathbf{B} \mathcal{A}_\mathsf{S}$	44.11	12.64
4	6	1636	7468	196	1924	23	$\mathcal{A}_\mathsf{P} \sqsubseteq_\mathbf{F} \mathcal{A}_\mathsf{S}$	30.28	10.39
							$\mathcal{A}_\mathsf{P} \sqsupseteq_\mathbf{B} \mathcal{A}_\mathsf{S}$	29.94	10.49

- Two-way inclusion could be checked for all parameters
  - $\mathcal{A}_P \sqsubseteq_\mathbf{F} \mathcal{A}_S$ and $\mathcal{A}_S \sqsubseteq_\mathbf{B} \mathcal{A}_P$ were found
- Space is serious problem
- Slower than implementation in [Kiefer et al. 2011]
  - ➡ Inclusion is harder to check than equivalence

undecidable [Blondel & Canterni, 2003]

P [Kiefer et al. 2011]

Experimental Results 2

- Verification of probable innocence of Crowds protocol
  [Konstantinos et al. 2006] [Reiter et al. 1998]
  - Probable innocence : a kind of anonymity

- Probable innocence can be proved by checking language inclusion [Hasuo et al. 2010]
  (not language equivalence)

Experimental Results 2

- Verification of probable innocence of Crowds protocol
  [Konstantinos et al. 2006] [Reiter et al. 1998]
  - Probable innocence : a kind of anonymity

- Probable innocence can be proved by checking language inclusion [Hasuo et al. 2010]
  (not language equivalence)

- We tried to prove for various parameters

Experimental Results 2

- Verification of probable innocence of Crowds protocol
  [Konstantinos et al. 2006] [Reiter et al. 1998]
  - Probable innocence : a kind of anonymity

- Probable innocence can be proved by checking language inclusion [Hasuo et al. 2010]
  (not language equivalence)

- We tried to prove for various parameters

➡ Simulation was not found for many parameters

Experimental Results 2

- Verification of probable innocence of Crowds protocol
  [Konstantinos et al. 2006] [Reiter et al. 1998]
  - Probable innocence : a kind of anonymity

- Probable innocence can be proved by checking language inclusion [Hasuo et al. 2010]
  (not language equivalence)

- We tried to prove for various parameters

➡ Simulation was not found for many parameters

➡ **fwd./ bwd. partial execution**

Experimental Results 2

param.			$\mathcal{A}_\mathsf{P}$		$\mathcal{A}_\mathsf{S}$			direction	time	space	$\boldsymbol{d}$
$\boldsymbol{n}$	$\boldsymbol{c}$	$\boldsymbol{p_f}$	#st.	#tr.	#st.	#tr.	$\boldsymbol{\lvert\Sigma\rvert}$	fwd./bwd.	(sec)	(GB)	
5	1	$\frac{\mathbf{9}}{\mathbf{10}}$	7	44	7	56	18	$\mathcal{A}_\mathsf{P}\sqsubseteq_{\mathbf{F}}\mathcal{A}_\mathsf{S}$	52.48	0.01	2
								$\mathcal{A}_\mathsf{P}\sqsubseteq_{\mathbf{B}}\mathcal{A}_\mathsf{S}$	0.01	0.01	2
7	1	$\frac{\mathbf{3}}{\mathbf{4}}$	9	88	9	118	26	$\mathcal{A}_\mathsf{P}\sqsubseteq_{\mathbf{F}}\mathcal{A}_\mathsf{S}$	0.15	0.03	2
								$\mathcal{A}_\mathsf{P}\sqsubseteq_{\mathbf{B}}\mathcal{A}_\mathsf{S}$	0.02	0.01	2
10	2	$\frac{\mathbf{4}}{\mathbf{5}}$	12	224	12	280	54	$\mathcal{A}_\mathsf{P}\sqsubseteq_{\mathbf{F}}\mathcal{A}_\mathsf{S}$	802.47	0.35	2
								$\mathcal{A}_\mathsf{P}\sqsubseteq_{\mathbf{B}}\mathcal{A}_\mathsf{S}$	0.05	0.03	2
20	6	$\frac{\mathbf{4}}{\mathbf{5}}$	22	1514	22	1696	238	$\mathcal{A}_\mathsf{P}\sqsubseteq_{\mathbf{F}}\mathcal{A}_\mathsf{S}$	T/O		2
								$\mathcal{A}_\mathsf{P}\sqsubseteq_{\mathbf{B}}\mathcal{A}_\mathsf{S}$	1.32	0.78	2
30	6	$\frac{\mathbf{4}}{\mathbf{5}}$	32	4732	32	5112	550	$\mathcal{A}_\mathsf{P}\sqsubseteq_{\mathbf{F}}\mathcal{A}_\mathsf{S}$	S/F		
								$\mathcal{A}_\mathsf{P}\sqsubseteq_{\mathbf{B}}\mathcal{A}_\mathsf{S}$	11.84	5.99	2

Experimental Results 2

param.			$\mathcal{A}_\mathsf{P}$		$\mathcal{A}_\mathsf{S}$			direction	time	space	d
n	c	p_f	#st.	#tr.	#st.	#tr.	$\lvert\boldsymbol{\Sigma}\rvert$	fwd./bwd.	(sec)	(GB)	
5	1	$\frac{9}{10}$	7	44	7	56	18	$\mathcal{A}_\mathsf{P}\sqsubseteq_\mathbf{F}\mathcal{A}_\mathsf{S}$	52.48	0.01	2
								$\mathcal{A}_\mathsf{P}\sqsubseteq_\mathbf{B}\mathcal{A}_\mathsf{S}$	0.01	0.01	2
7	1	$\frac{3}{4}$	9	88	9	118	26	$\mathcal{A}_\mathsf{P}\sqsubseteq_\mathbf{F}\mathcal{A}_\mathsf{S}$	0.15	0.03	2
								$\mathcal{A}_\mathsf{P}\sqsubseteq_\mathbf{B}\mathcal{A}_\mathsf{S}$	0.02	0.01	2
10	2	$\frac{4}{5}$	12	224	12	280	54	$\mathcal{A}_\mathsf{P}\sqsubseteq_\mathbf{F}\mathcal{A}_\mathsf{S}$	802.47	0.35	2
								$\mathcal{A}_\mathsf{P}\sqsubseteq_\mathbf{B}\mathcal{A}_\mathsf{S}$	0.05	0.03	2
20	6	$\frac{4}{5}$	22	1514	22	1696	238	$\mathcal{A}_\mathsf{P}\sqsubseteq_\mathbf{F}\mathcal{A}_\mathsf{S}$	T/O		2
								$\mathcal{A}_\mathsf{P}\sqsubseteq_\mathbf{B}\mathcal{A}_\mathsf{S}$	1.32	0.78	2
30	6	$\frac{4}{5}$	32	4732	32	5112	550	$\mathcal{A}_\mathsf{P}\sqsubseteq_\mathbf{F}\mathcal{A}_\mathsf{S}$	S/F		
								$\mathcal{A}_\mathsf{P}\sqsubseteq_\mathbf{B}\mathcal{A}_\mathsf{S}$	11.84	5.99	2

- Simulation finally found for many parameters

Experimental Results 2

param.			$\mathcal{A}_{\mathsf{P}}$		$\mathcal{A}_{\mathsf{S}}$			direction	time	space	d
n	c	p_f	#st.	#tr.	#st.	#tr.	$\lvert\boldsymbol{\Sigma}\rvert$	fwd./bwd.	(sec)	(GB)	
5	1	$\frac{9}{10}$	7	44	7	56	18	$\mathcal{A}_{\mathsf{P}} \sqsubseteq_{\mathbf{F}} \mathcal{A}_{\mathsf{S}}$	52.48	0.01	2
								$\mathcal{A}_{\mathsf{P}} \sqsubseteq_{\mathbf{B}} \mathcal{A}_{\mathsf{S}}$	0.01	0.01	2
7	1	$\frac{3}{4}$	9	88	9	118	26	$\mathcal{A}_{\mathsf{P}} \sqsubseteq_{\mathbf{F}} \mathcal{A}_{\mathsf{S}}$	0.15	0.03	2
								$\mathcal{A}_{\mathsf{P}} \sqsubseteq_{\mathbf{B}} \mathcal{A}_{\mathsf{S}}$	0.02	0.01	2
10	2	$\frac{4}{5}$	12	224	12	280	54	$\mathcal{A}_{\mathsf{P}} \sqsubseteq_{\mathbf{F}} \mathcal{A}_{\mathsf{S}}$	802.47	0.35	2
								$\mathcal{A}_{\mathsf{P}} \sqsubseteq_{\mathbf{B}} \mathcal{A}_{\mathsf{S}}$	0.05	0.03	2
20	6	$\frac{4}{5}$	22	1514	22	1696	238	$\mathcal{A}_{\mathsf{P}} \sqsubseteq_{\mathbf{F}} \mathcal{A}_{\mathsf{S}}$	T/O		2
								$\mathcal{A}_{\mathsf{P}} \sqsubseteq_{\mathbf{B}} \mathcal{A}_{\mathsf{S}}$	1.32	0.78	2
30	6	$\frac{4}{5}$	32	4732	32	5112	550	$\mathcal{A}_{\mathsf{P}} \sqsubseteq_{\mathbf{F}} \mathcal{A}_{\mathsf{S}}$	S/F		
								$\mathcal{A}_{\mathsf{P}} \sqsubseteq_{\mathbf{B}} \mathcal{A}_{\mathsf{S}}$	11.84	5.99	2

- Simulation finally found for many parameters
- Space is serious problem

Experimental Results 2

param.			$\mathcal{A}_{\mathsf{P}}$		$\mathcal{A}_{\mathsf{S}}$			direction	time	space	d
n	c	p_f	#st.	#tr.	#st.	#tr.	$\lvert\Sigma\rvert$	fwd./bwd.	(sec)	(GB)	
5	1	$\frac{9}{10}$	7	44	7	56	18	$\mathcal{A}_{\mathsf{P}} \sqsubseteq_{\mathbf{F}} \mathcal{A}_{\mathsf{S}}$	52.48	0.01	2
								$\mathcal{A}_{\mathsf{P}} \sqsubseteq_{\mathbf{B}} \mathcal{A}_{\mathsf{S}}$	0.01	0.01	2
7	1	$\frac{3}{4}$	9	88	9	118	26	$\mathcal{A}_{\mathsf{P}} \sqsubseteq_{\mathbf{F}} \mathcal{A}_{\mathsf{S}}$	0.15	0.03	2
								$\mathcal{A}_{\mathsf{P}} \sqsubseteq_{\mathbf{B}} \mathcal{A}_{\mathsf{S}}$	0.02	0.01	2
10	2	$\frac{4}{5}$	12	224	12	280	54	$\mathcal{A}_{\mathsf{P}} \sqsubseteq_{\mathbf{F}} \mathcal{A}_{\mathsf{S}}$	802.47	0.35	2
								$\mathcal{A}_{\mathsf{P}} \sqsubseteq_{\mathbf{B}} \mathcal{A}_{\mathsf{S}}$	0.05	0.03	2
20	6	$\frac{4}{5}$	22	1514	22	1696	238	$\mathcal{A}_{\mathsf{P}} \sqsubseteq_{\mathbf{F}} \mathcal{A}_{\mathsf{S}}$	T/O		2
								$\mathcal{A}_{\mathsf{P}} \sqsubseteq_{\mathbf{B}} \mathcal{A}_{\mathsf{S}}$	1.32	0.78	2
30	6	$\frac{4}{5}$	32	4732	32	5112	550	$\mathcal{A}_{\mathsf{P}} \sqsubseteq_{\mathbf{F}} \mathcal{A}_{\mathsf{S}}$	S/F		
								$\mathcal{A}_{\mathsf{P}} \sqsubseteq_{\mathbf{B}} \mathcal{A}_{\mathsf{S}}$	11.84	5.99	2

- Simulation finally found for many parameters
- Space is serious problem
- Bwd. simulation is much faster than fwd. simulation
    - ➡ Due to peculiar shape of automaton?

Overview

1. Matrix Simulation
   - Motivation
   - Semiring-Weighted Automaton and Matrix Simulation
   - Origin: from Theory of Coalgebra
2. Partial Execution (to be More “Complete”)
3. Specific Examples
   - Example 1 : $\mathcal{S}_{+,\times}$-weighted Automaton
   - Example 2 : $\boldsymbol{\mathcal{S}}_{\mathbf{max},+}$-weighted Automaton
4. Conclusion and Future Works

Comparison with Other Simulations for $\mathcal{S}_{\mathbf{max},+}$-Weighted Automata

$$\mathcal{S}_{\mathbf{max},+} = ([-\infty, \infty], \mathbf{max}, -\infty, +, \mathbf{0}, \leq)$$

- Simulation by Chatterjee et al. (2010) (G-simulation)
  - Game-theoretic simulation
  - Simulation for automata for infinite-length words
  - Easy to modify for automata for finite-length words

Thm: If $\mathcal{A}$ has no trap states
(i.e. every states can reach the final state),

$$\mathcal{A} \sqsubseteq_{\mathbf{F}} \mathcal{B} \;\Leftrightarrow\; \mathcal{A} \sqsubseteq_{\mathbf{G}} \mathcal{B}$$

Experimental Result

- Algorithm for linear inequalities on $\mathcal{S}_{\max,+}$ is introduced by Butkovic et al.

  → We implemented using the algorithm

  [Butkovic et al. 2006]

Experimental Result

- Algorithm for linear inequalities on $\mathcal{S}_{\max,+}$ is introduced by Butkovic et al.

  ⇒ We implemented using the algorithm [Butkovic et al. 2006]

- We could not find good benchmark

  ⇒ Test $\max +$-sim for $\left\{\begin{array}{ll} \mathcal{A} \sqsubseteq_{F/B} \mathcal{A} & \text{(almost always yes)} \\ \mathcal{A} \sqsubseteq_{F/B} \mathcal{B} & \text{(almost always no)} \end{array}\right.$

  ($\mathcal{A}, \mathcal{B}$: randomly generated automata)

Experimental Result

- Algorithm for linear inequalities on $\mathcal{S}_{\max,+}$ is introduced by Butkovic et al.

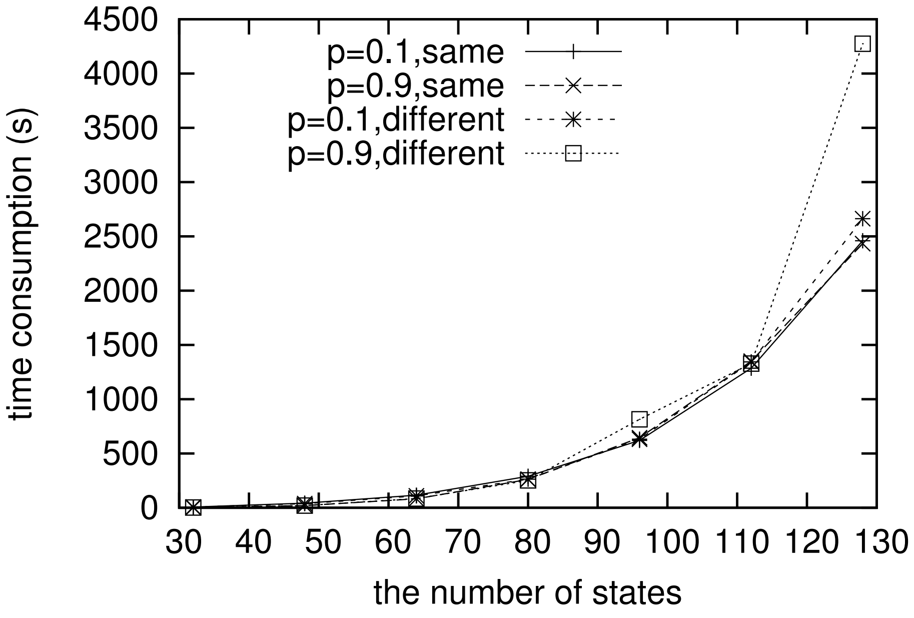
  We implemented using the algorithm

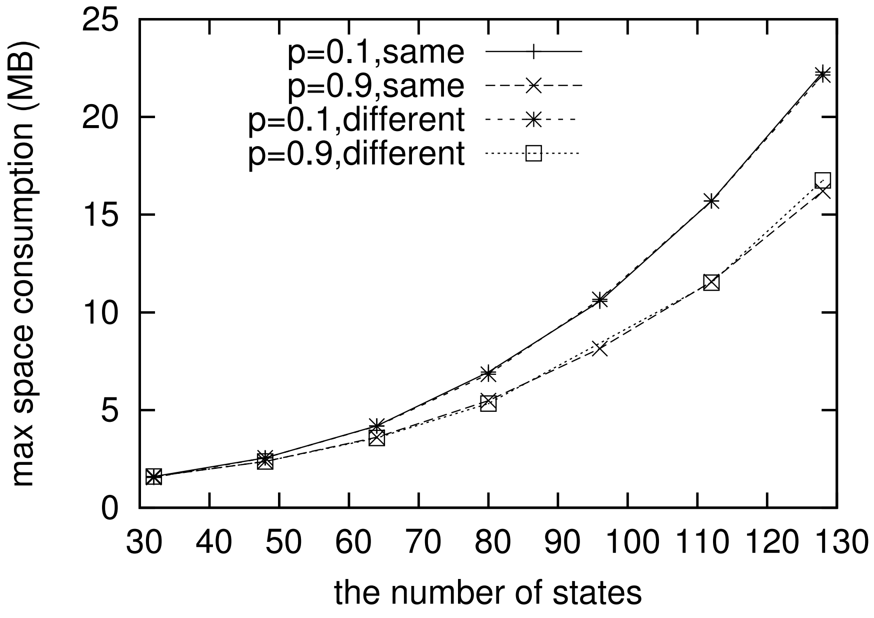
  [Butkovic et al. 2006]

- We could not find good benchmark

$$\text{Test } \max + \text{-sim for } \begin{cases} \mathcal{A} \sqsubseteq_{F/B} \mathcal{A} & \text{(almost always yes)} \\ \mathcal{A} \sqsubseteq_{F/B} \mathcal{B} & \text{(almost always no)} \end{cases}$$

($\mathcal{A}, \mathcal{B}$: randomly generated automata)





Experimental Result

- Algorithm for linear inequalities on $\mathcal{S}_{\max,+}$ is introduced by Butkovic et al.

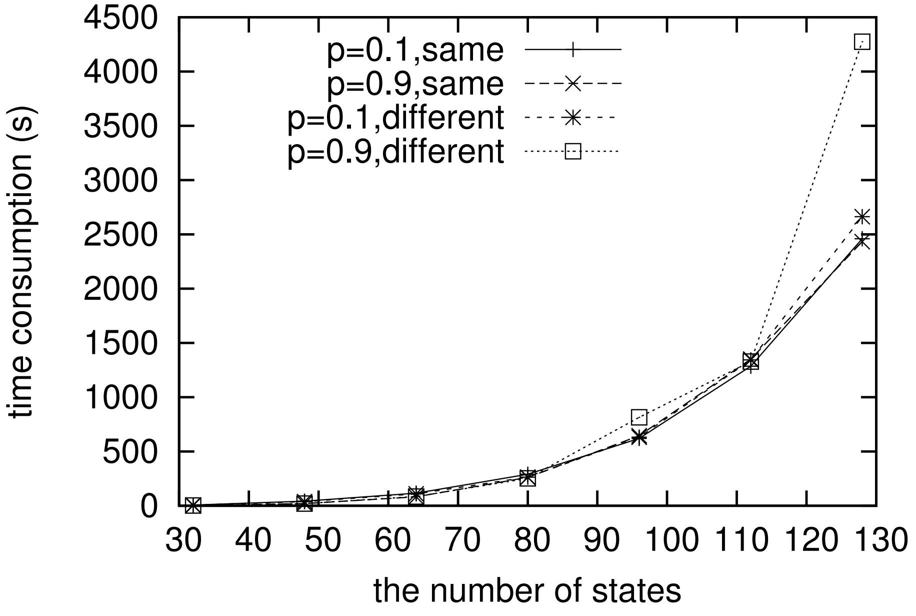
  We implemented using the algorithm

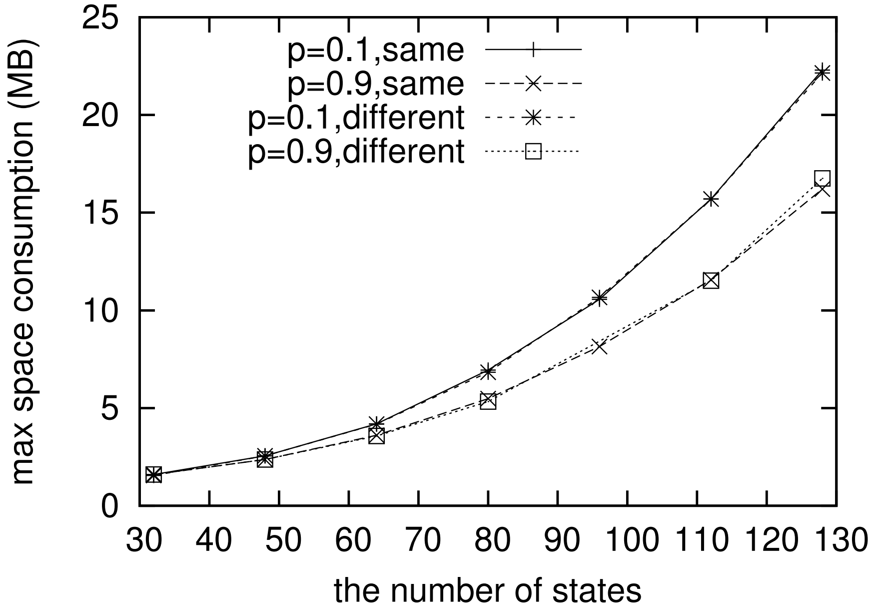
  [Butkovic et al. 2006]

- We could not find good benchmark

  Test $\max +$-sim for $\begin{cases} \mathcal{A} \sqsubseteq_{F/B} \mathcal{A} & \text{(almost always yes)} \\ \mathcal{A} \sqsubseteq_{F/B} \mathcal{B} & \text{(almost always no)} \end{cases}$

  ($\mathcal{A}, \mathcal{B}$: randomly generated automata)





Both increases non-linearly

Room for optimization?

Overview

1. Matrix Simulation
   - Motivation
   - Semiring-Weighted Automaton and Matrix Simulation
   - Origin: from Theory of Coalgebra
2. Partial Execution (to be More “Complete”)
3. Specific Examples
   - Example 1 : $\mathcal{S}_{+,\times}$-weighted Automaton
   - Example 2 : $\mathcal{S}_{\mathbf{max},+}$-weighted Automaton
4. Conclusion and Future Works

Conclusion

- Matrix-based simulation (**matrix simulation**) to prove language inclusion between weighted automata
- Transformation of automata (**partial execution**) that increases matrix simulations
- Study for specific semirings — $\mathcal{S}_{+,\times}$ and $\mathcal{S}_{\mathbf{max},+}$

Future Works

- Matrix simulation can be defined for other transition types by its generality
  - ➡ Change $\boldsymbol{F}$ to other polynomial functors

    e.g. $\boldsymbol{F} = \mathbf{1} + \mathbf{\Sigma} \times (\_) \times (\_)$

    (Automaton that accepts trees)

- Matrix simulation for automaton for infinite-length words

- Optimization of implementation