Learning Probabilistic Systems from Tree Samples

Anvesh Komuravelli

Computer Science Department
Carnegie Mellon University

June 28, 2012

Joint work with Corina S. Păsăreanu and Edmund M. Clarke

The Problem

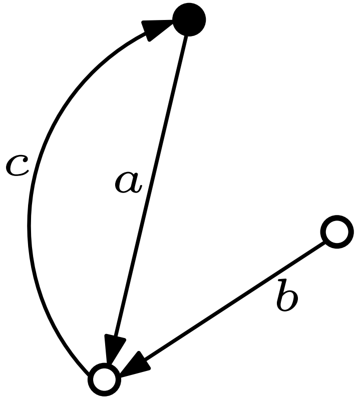
$$L \overset{?}{\preceq} P$$

Implementation Specification

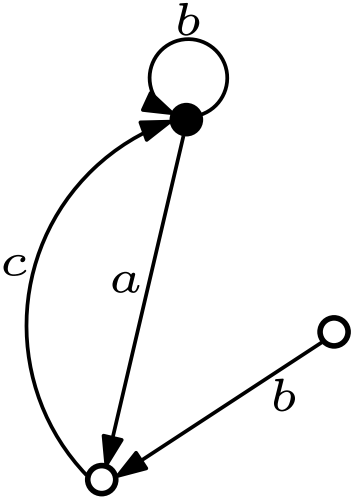
Labeled Transition System (LTS)

Labeled Transition System
(LTS)

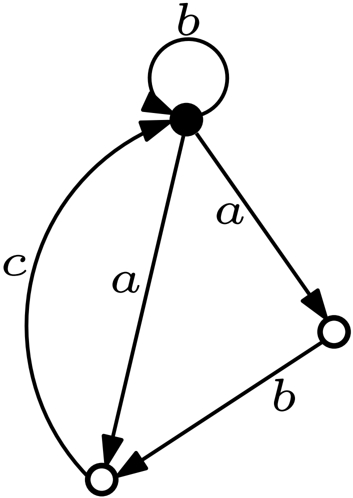
Labeled Transition System
(LTS)



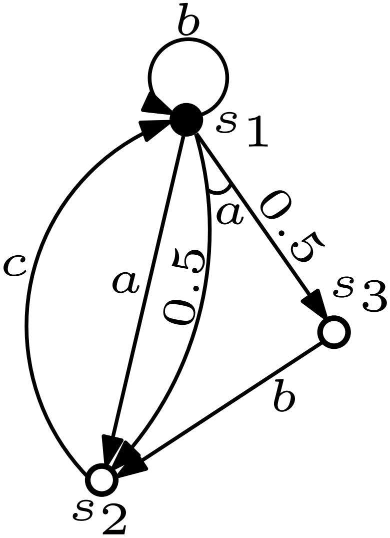
Labeled Transition System
(LTS)



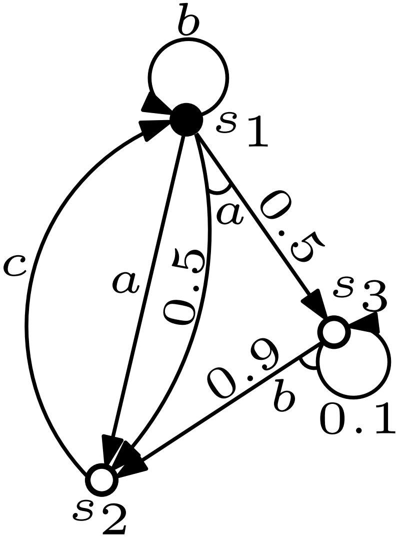
Labeled Transition System
(LTS)



Labeled *Probabilistic* Transition System (LPTS)



Labeled *Probabilistic* Transition System (LPTS)



The Problem

Carnegie Mellon

$$L \overset{?}{\preceq} P$$

Implementation Specification

The Problem

$$L_1 \parallel L_2 \parallel \cdots \parallel L_n \quad \overset{?}{\preceq} \quad P$$

Implementation Specification

The Problem

$$L_1 \parallel L_2 \parallel \cdots \parallel L_n \overset{?}{\preceq} P$$

Implementation Specification

State-space Explosion!

Assume-Guarantee Reasoning

$$\frac{\begin{array}{c} L_2 \preceq A \\ L_1 \parallel A \preceq P \end{array}}{L_1 \parallel L_2 \preceq P}$$

Assume-Guarantee Reasoning

$$\frac{\begin{array}{c} L_2 \preceq A \\ L_1 \parallel A \preceq P \end{array}}{L_1 \parallel L_2 \preceq P}$$

How to automatically infer *A*?

Prior Work

Carnegie Mellon

- If $L \not\preceq P$, a stochastic tree counterexample C can be found such that $C \preceq L$ but $C \not\preceq P$ [Komuravelli et al. CAV 2012].

- If $L \npreceq P$, a stochastic tree counterexample C can be found such that $C \preceq L$ but $C \npreceq P$ [Komuravelli et al. CAV 2012].
- For LTSes, Angluin-style active learning algorithm [Angluin 1987] was used to learn Tree Automata [Chaki et al. 2005].

Prior Work

- If $L \not\preceq P$, a stochastic tree counterexample C can be found such that $C \preceq L$ but $C \not\preceq P$ [Komuravelli et al. CAV 2012].
- For LTSes, Angluin-style active learning algorithm [Angluin 1987] was used to learn Tree Automata [Chaki et al. 2005].
- Not aware of probabilistic variant of tree automata.

Incremental Approach

$$\frac{\begin{array}{c} L_2 \preceq A \\ L_1 \parallel A \preceq P \end{array}}{L_1 \parallel L_2 \preceq P}$$

Incremental Approach

$$\frac{L_2 \preceq A \qquad L_1 \parallel A \preceq P}{L_1 \parallel L_2 \preceq P} \xrightarrow{\textcolor{red}{\mathrm{F}}} C \preceq L_2, C \not\preceq A$$

Incremental Approach

$$\frac{L_2 \preceq A \qquad L_1 \parallel A \preceq P}{L_1 \parallel L_2 \preceq P} \xrightarrow{\text{F}} C \preceq L_2, C \not\preceq A$$

positive

Incremental Approach

$$\frac{\begin{array}{c} L_2 \preceq A \\ L_1 \parallel A \preceq P \end{array}}{L_1 \parallel L_2 \preceq P} \xrightarrow{\mathrm{F}} C \preceq L_1 \parallel A, C \not\preceq P$$

Incremental Approach

$$\frac{\begin{array}{c} L_2 \preceq A \\ L_1 \parallel A \preceq P \end{array}}{L_1 \parallel L_2 \preceq P} \xrightarrow{\color{red}{\mathrm{F}}} C_A \preceq A, C_A \not\preceq P$$

Incremental Approach

$$\frac{\begin{array}{c} L_2 \preceq A \\ L_1 \parallel A \preceq P \end{array}}{L_1 \parallel L_2 \preceq P} \xrightarrow{\color{red}{\mathrm{F}}} C_A \preceq A, C_A \not\preceq P$$

negative

Incremental Approach

Carnegie Mellon

$$\frac{L_2 \preceq A \qquad L_1 \parallel A \preceq P}{L_1 \parallel L_2 \preceq P}$$

Learning an unknown LPTS using positive and negative counterexamples!

Similar to that of an existing work for DFAs and trace counterexamples [Gupta et al. 2007].

Our Goals

Question 1 How to automatically infer a H consistent with all the counterexamples?

Our Goals

Question 1 How to automatically infer a H consistent with all the counterexamples?

That is, how to find H such that

- for every $P \in \mathcal{P}$, $P \preceq H$ and
- for every $N \in \mathcal{N}$, $N \not\preceq H$?

Our Goals

Question 1 How to automatically infer a H consistent with all the counterexamples?

That is, how to find H such that

- for every $P \in \mathcal{P}$, $P \preceq H$ and
- for every $N \in \mathcal{N}$, $N \not\preceq H$?

Question 2 Will the sequence of hypotheses converge?

Our Goals

Carnegie Mellon

Question 1 How to automatically infer a H consistent with all the counterexamples?

That is, how to find H such that

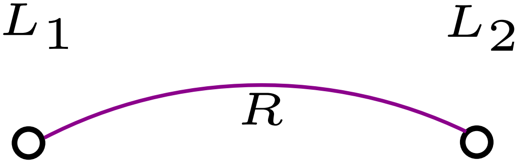
- for every $P \in \mathcal{P}$, $P \preceq H$ and
- for every $N \in \mathcal{N}$, $N \not\preceq H$?

Question 2 Will the sequence of hypotheses converge?

Learning Probabilistic Systems from Tree Samples!

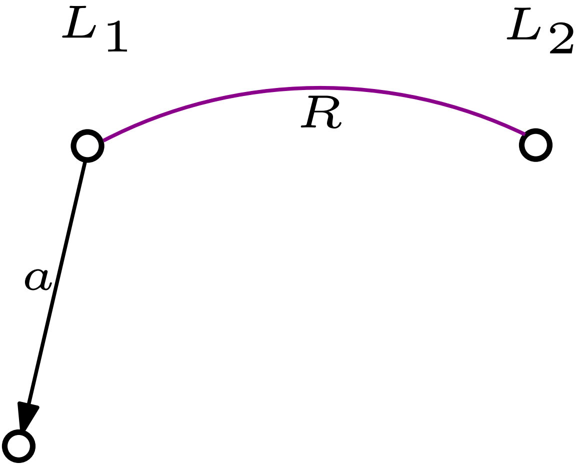
Strong Simulation for LTSes

[Milner 1971]

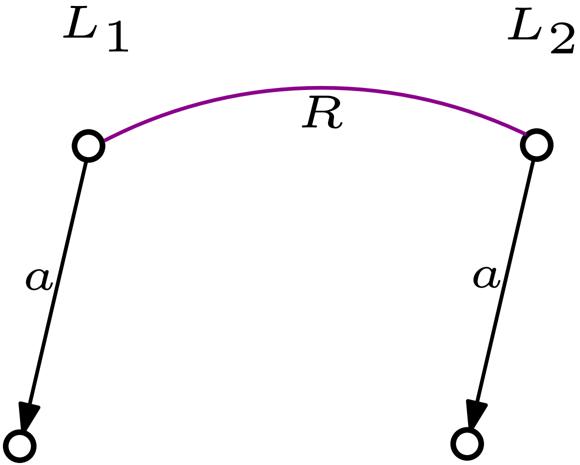


Strong Simulation for LTSes

[Milner 1971]

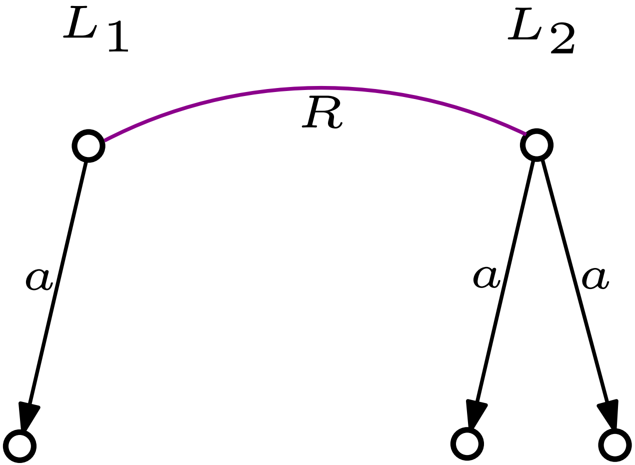


Strong Simulation for LTSes
[Milner 1971]

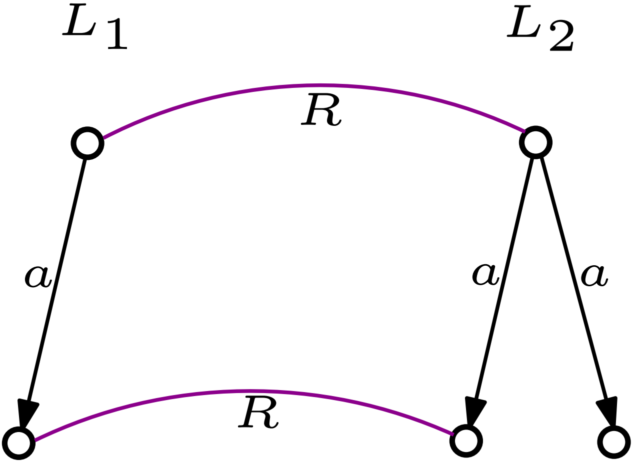


Strong Simulation for LTSes

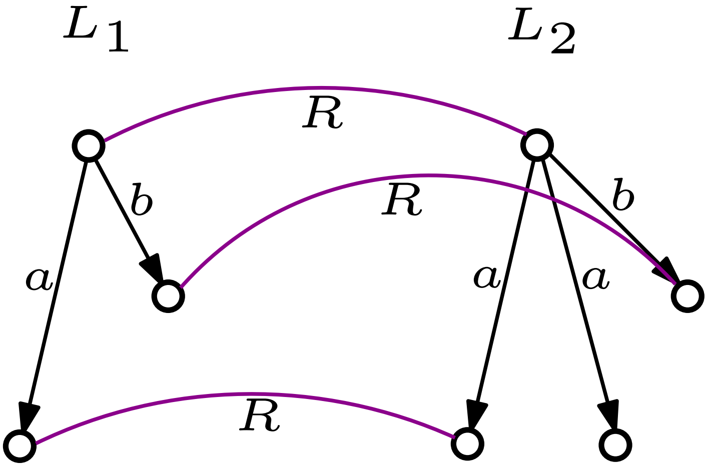
[Milner 1971]



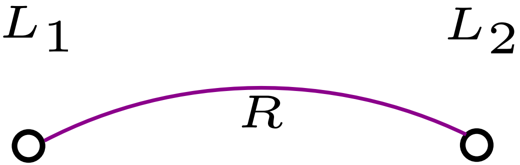
Strong Simulation for LTSes
[Milner 1971]



Strong Simulation for LTSes
[Milner 1971]

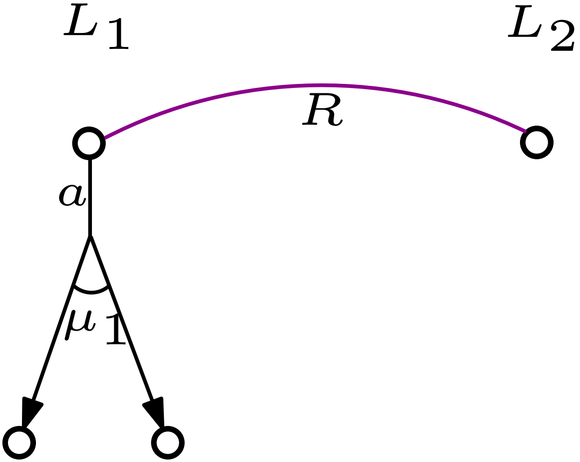


Strong Simulation for LPTSes
[Segala and Lynch 1995]



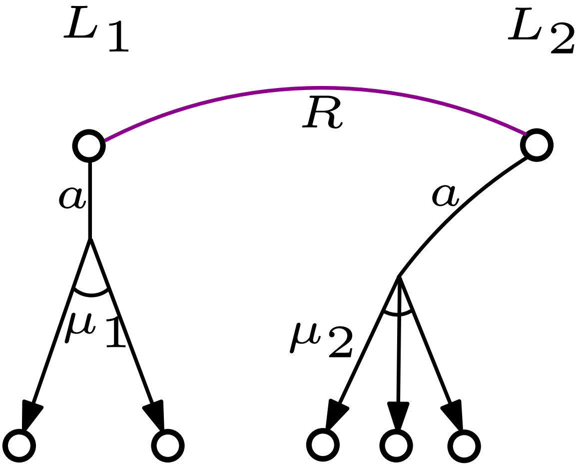
Strong Simulation for LPTSes

[Segala and Lynch 1995]



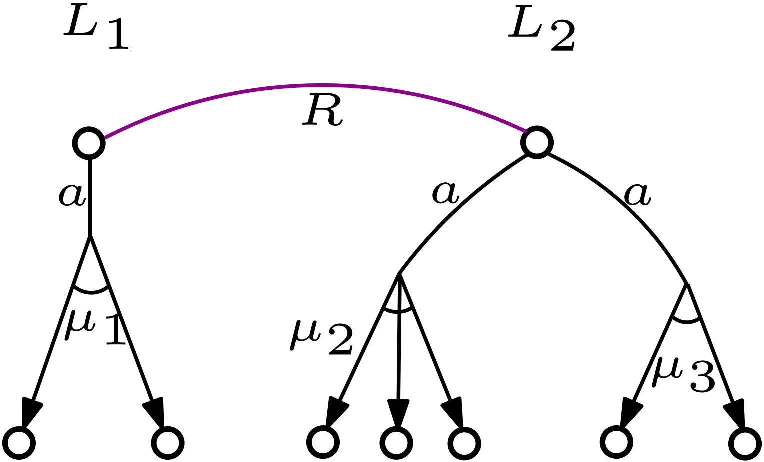
Strong Simulation for LPTSes

[Segala and Lynch 1995]



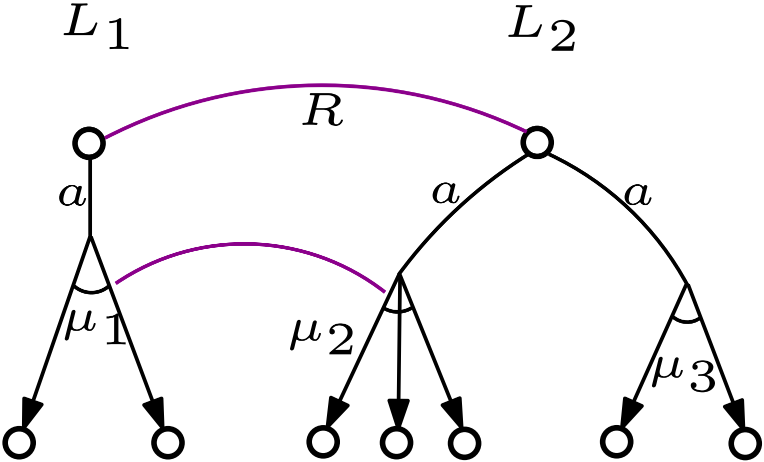
Strong Simulation for LPTSes

[Segala and Lynch 1995]



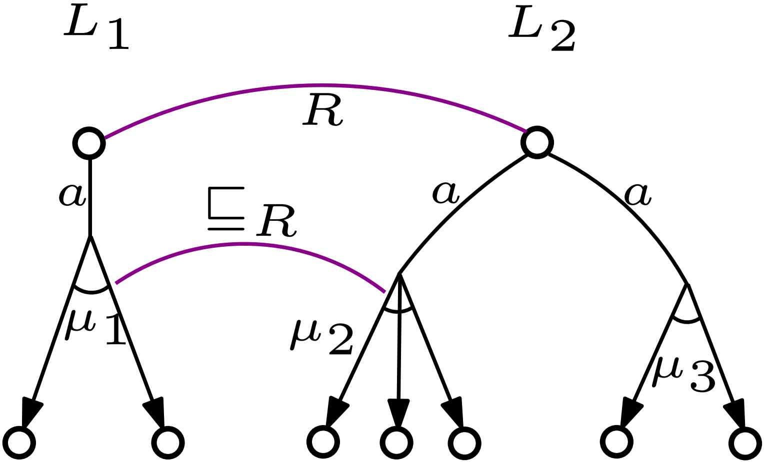
Strong Simulation for LPTSes

[Segala and Lynch 1995]

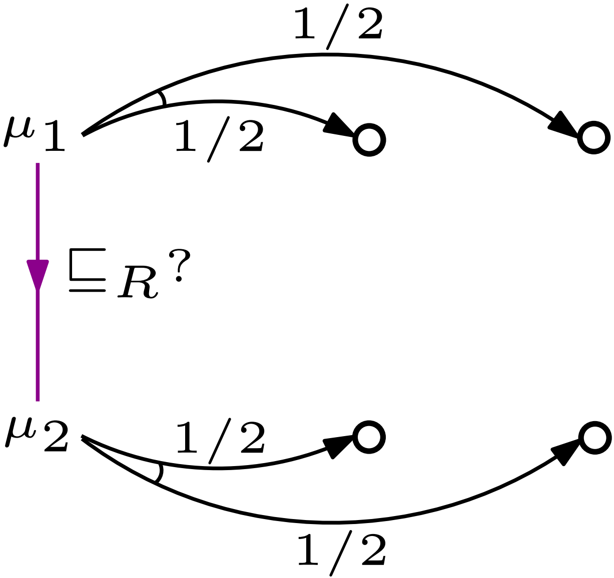


Strong Simulation for LPTSes

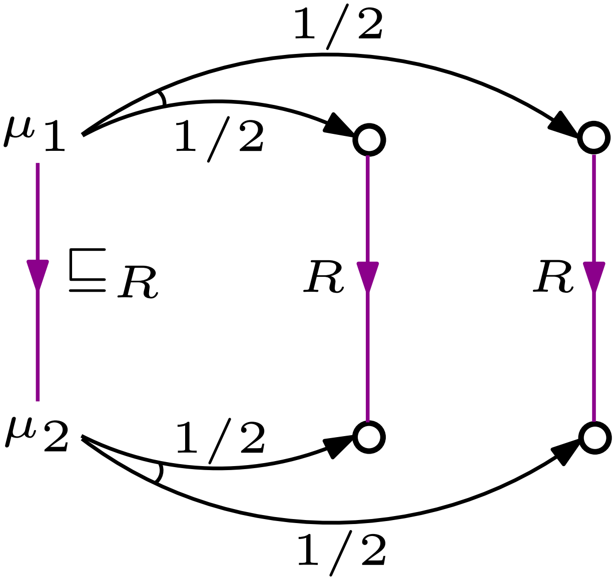
[Segala and Lynch 1995]



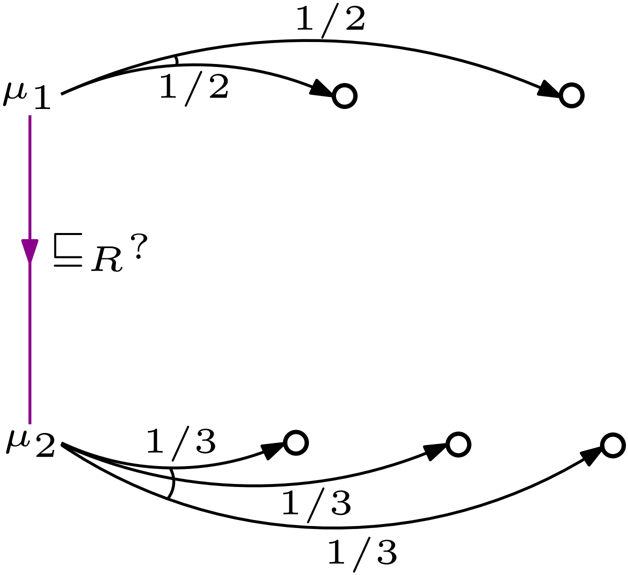
$\sqsubseteq_R$ as Matching the Probabilities
[Segala and Lynch 1995]



$\sqsubseteq_R$ as Matching the Probabilities
[Segala and Lynch 1995]



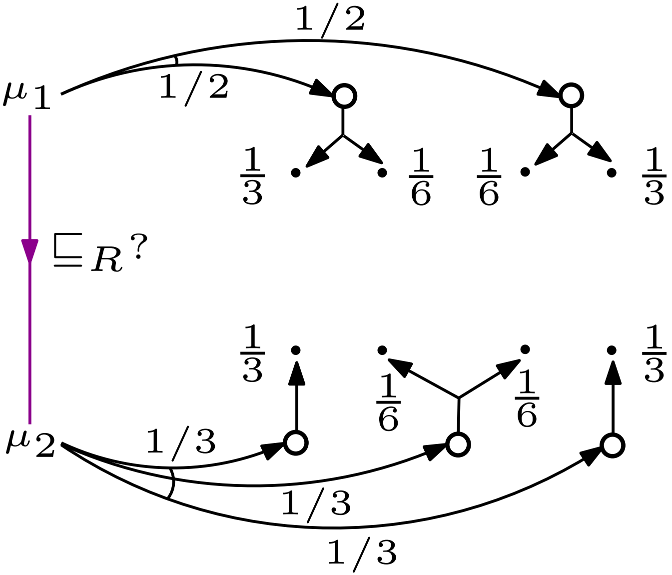
$\sqsubseteq_R$ as Matching the Probabilities
[Segala and Lynch 1995]



$\sqsubseteq_R$ as Matching the Probabilities

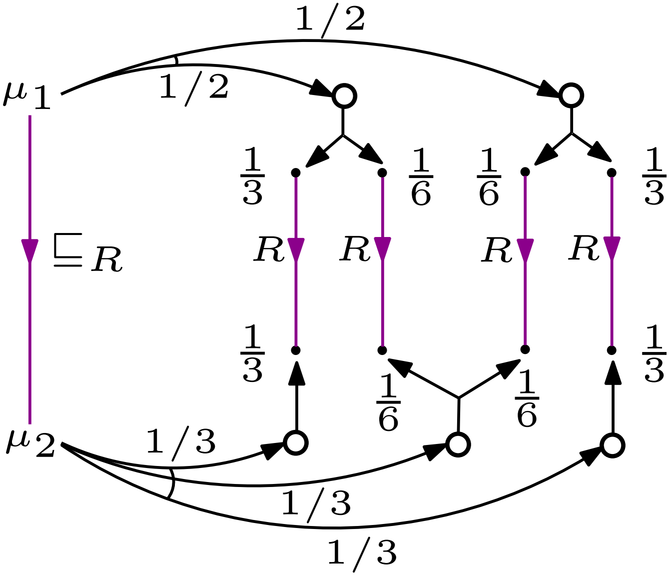
[Segala and Lynch 1995]

Carnegie Mellon



$\sqsubseteq_R$ as Matching the Probabilities

[Segala and Lynch 1995]



The Preorder $\preceq$

- $s_1 \preceq s_2$ iff there is a strong simulation R with $s_1 R s_2$.
- $\preceq$ is a preorder (reflexive and transitive).

The Preorder $\preceq$

- $s_1 \preceq s_2$ iff there is a strong simulation R with $s_1 R s_2$.
- $\preceq$ is a preorder (reflexive and transitive).
- $L_1 \preceq L_2$ iff $s_1^0 \preceq s_2^0$.

The Preorder $\preceq$

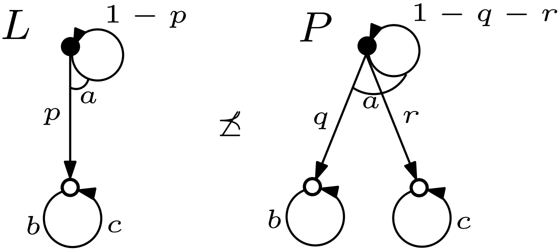
Carnegie Mellon

- $s_1 \preceq s_2$ iff there is a strong simulation R with $s_1 R s_2$.
- $\preceq$ is a preorder (reflexive and transitive).
- $L_1 \preceq L_2$ iff $s_1^0 \preceq s_2^0$.
- Decidable in poly-time [Baier et al. 2000, Zhang 2008].

When $L \preceq P$ fails...

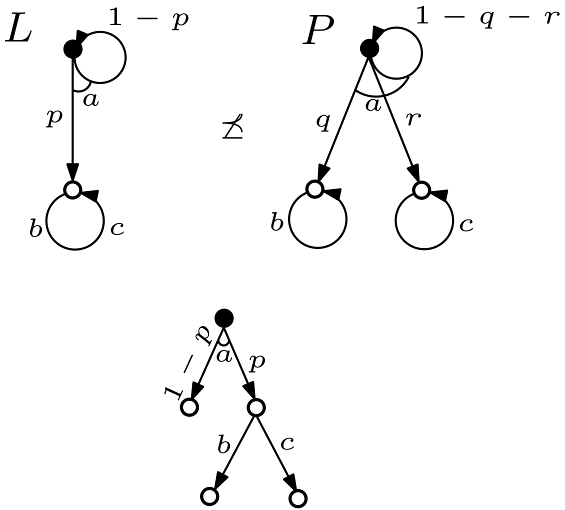
Carnegie Mellon

A tree counterexample C can be found such that $C \preceq L$ and $C \not\preceq P$ [Komuravelli et al. CAV 2012].



When $L \preceq P$ fails...

A tree counterexample C can be found such that $C \preceq L$ and $C \not\preceq P$ [Komuravelli et al. CAV 2012].

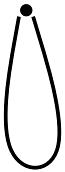


Our Goals

Question 1 How to automatically infer a H consistent with all the counterexamples?

Question 2 Will the sequence of hypotheses converge?

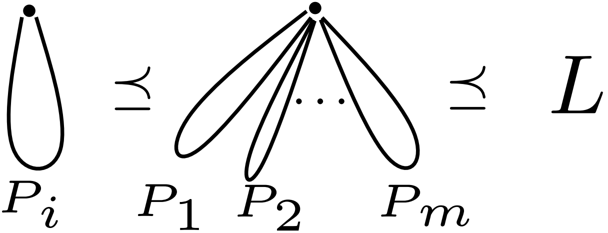
Learning a Consistent LPTS



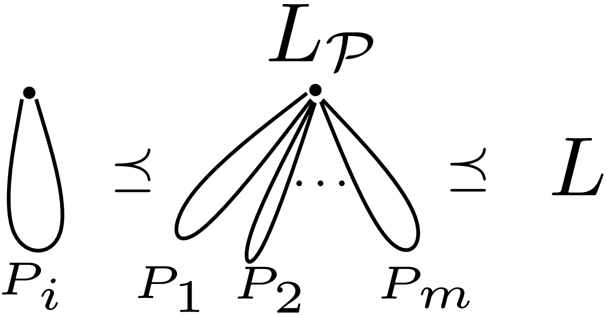
P_i

$\preceq \quad L$

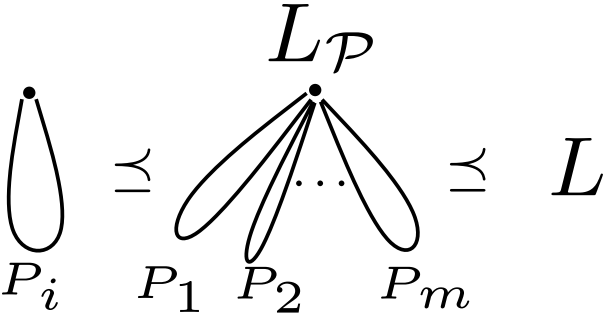
Learning a Consistent LPTS



Learning a Consistent LPTS

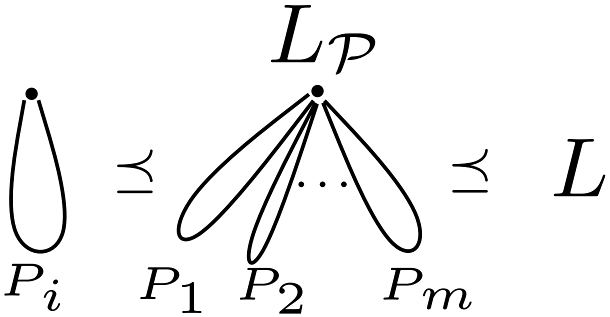


Learning a Consistent LPTS

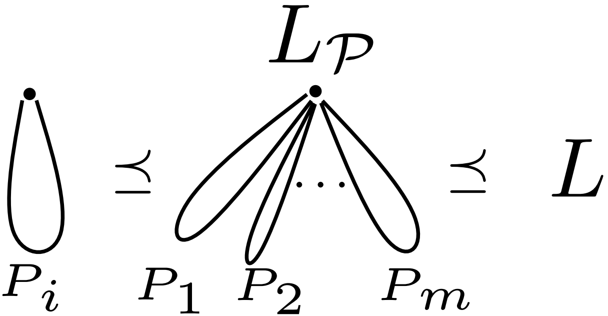


$$N \npreceq L \Longrightarrow N \npreceq L_{\mathcal{P}}$$

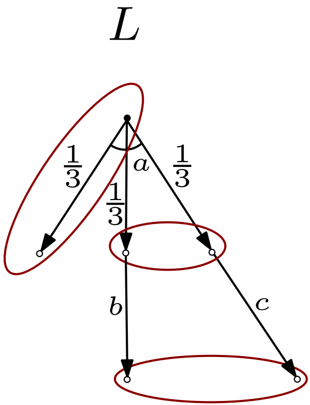
Learning a Consistent LPTS



$H = L_{\mathcal{P}}$?

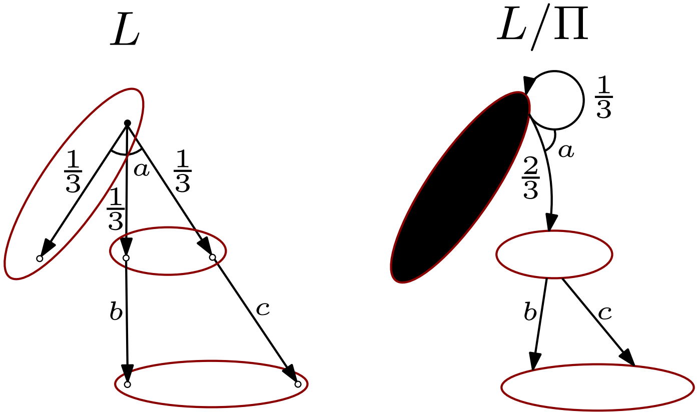


$H = L_{\mathcal{P}}$? NO!



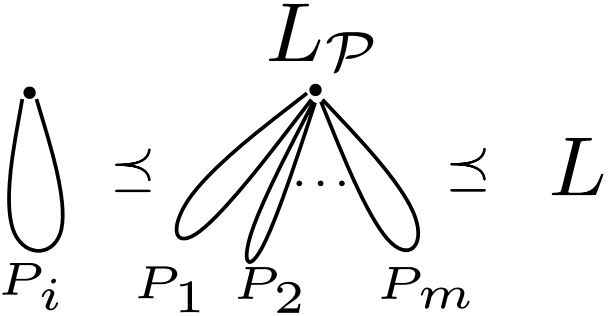
L
$\frac{1}{3}$
a
$\frac{1}{3}$
$\frac{1}{3}$
b
c

Partitions and Quotients



Partition-based Approach

Recall...



Partition-based Approach

- $L_{\mathcal{P}} \preceq L$.

Partition-based Approach

Carnegie Mellon

- $L_{\mathcal{P}} \preceq L$. Let R be a strong simulation between them.

Partition-based Approach

- $L_{\mathcal{P}} \preceq L$. Let R be a strong simulation between them.
- Consider $s_1 T s_2$ iff $R(s_1) = R(s_2)$.
- T induces a partition Π.

Partition-based Approach

- $L_{\mathcal{P}} \preceq L$. Let R be a strong simulation between them.
- Consider $s_1 T s_2$ iff $R(s_1) = R(s_2)$.
- T induces a partition Π.
- $L_{\mathcal{P}} \preceq L_{\mathcal{P}}/\Pi \preceq L$.

Partition-based Approach

- $L_{\mathcal{P}} \preceq L$. Let R be a strong simulation between them.
- Consider $s_1 T s_2$ iff $R(s_1) = R(s_2)$.
- T induces a partition Π.
- $L_{\mathcal{P}} \preceq L_{\mathcal{P}}/\Pi \preceq L$.
- Exponentially many equivalence classes.

Partition-based Approach

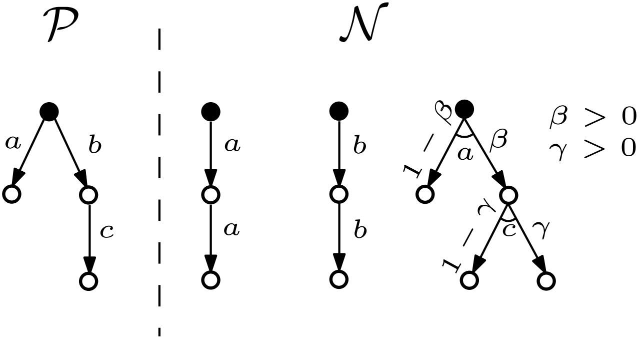
Carnegie Mellon

The Algorithm Find the *least-sized* partition Π such that $L_{\mathcal{P}}/\Pi$ is consistent!

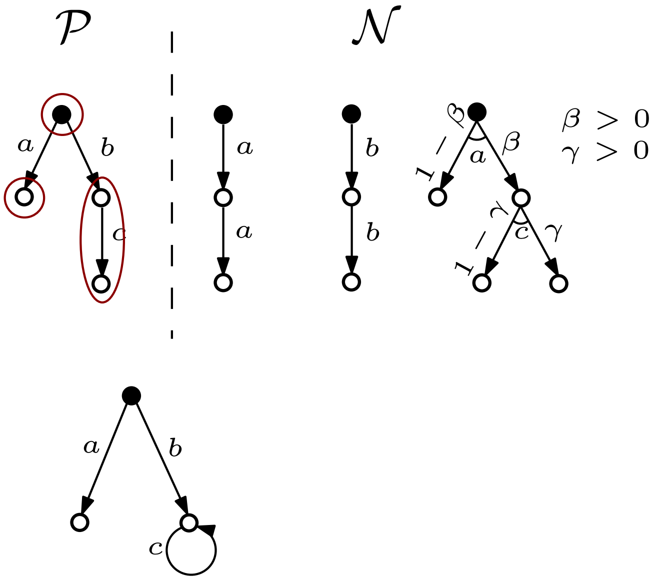
The Algorithm Find the *least-sized* partition Π such that $L_{\mathcal{P}}/\Pi$ is consistent!

Can be exponentially worse than the best.

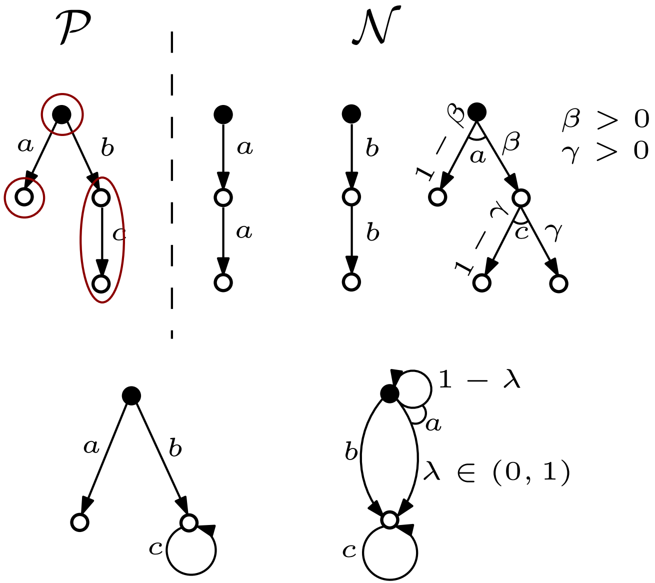
How good are Partitions?



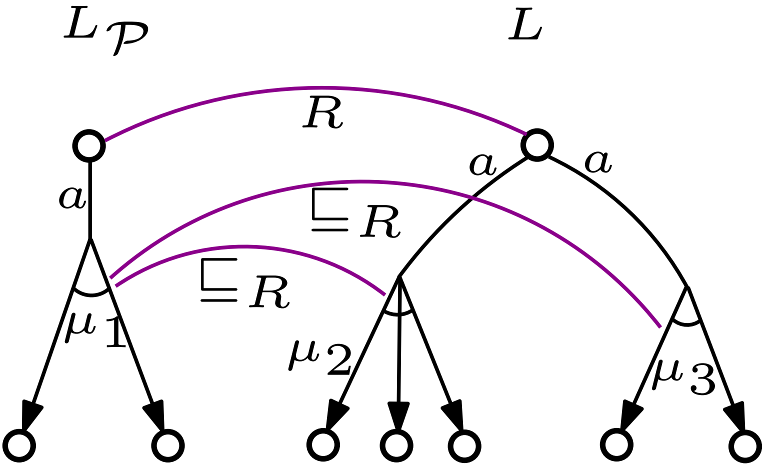
How good are Partitions?



How good are Partitions?

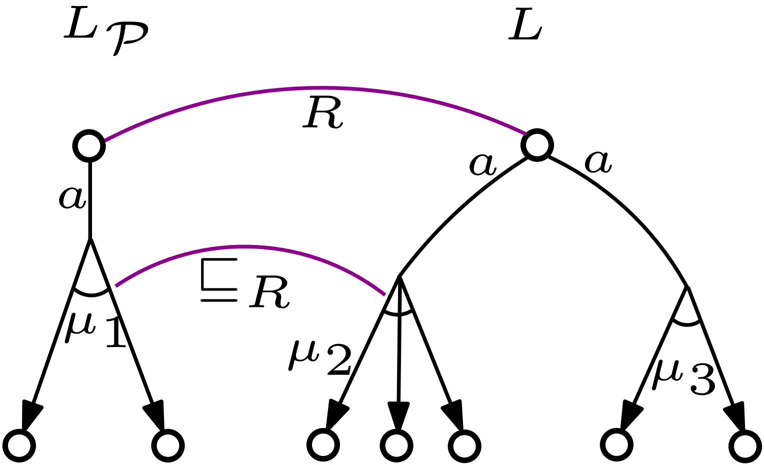


$$L_{\mathcal{P}} \preceq H \preceq L.$$

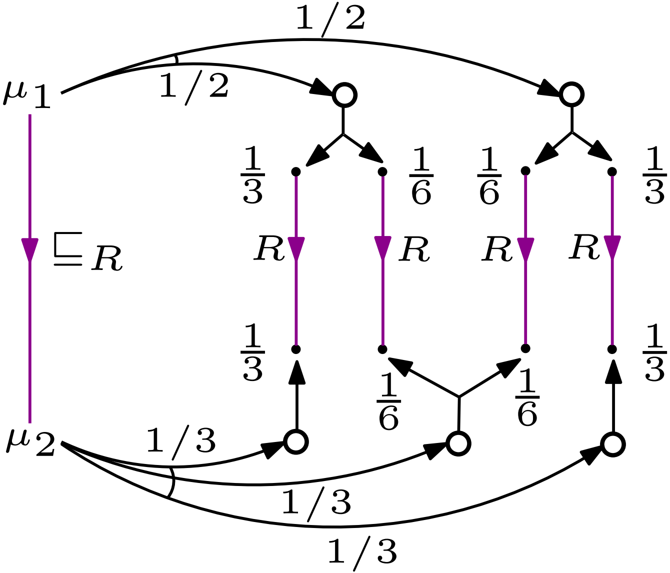


$L_{\mathcal{P}}$
L
R
a
a
a
$\sqsubseteq_R$
$\sqsubseteq_R$
μ_1
μ_2
μ_3

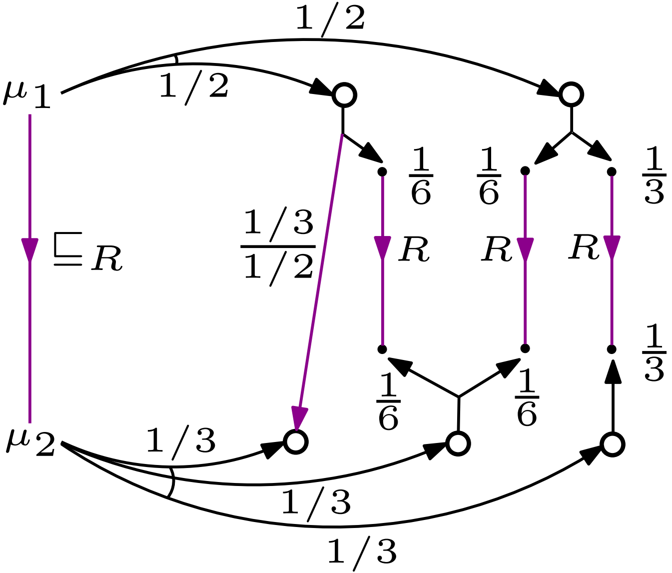
Towards being Optimal



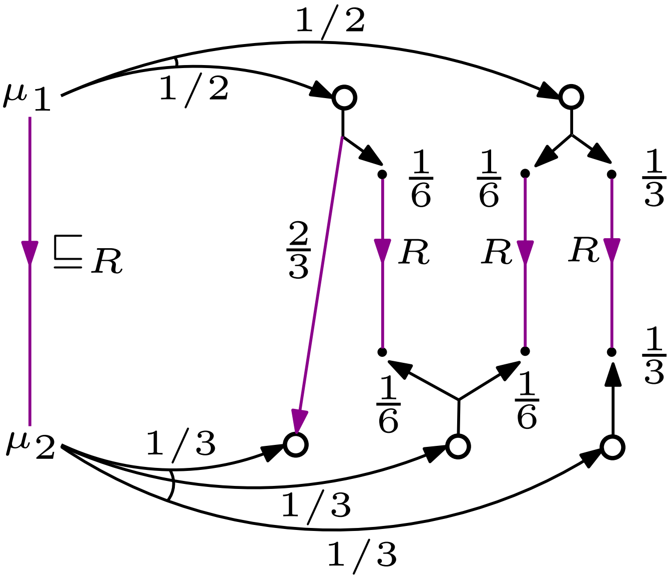
Towards being Optimal



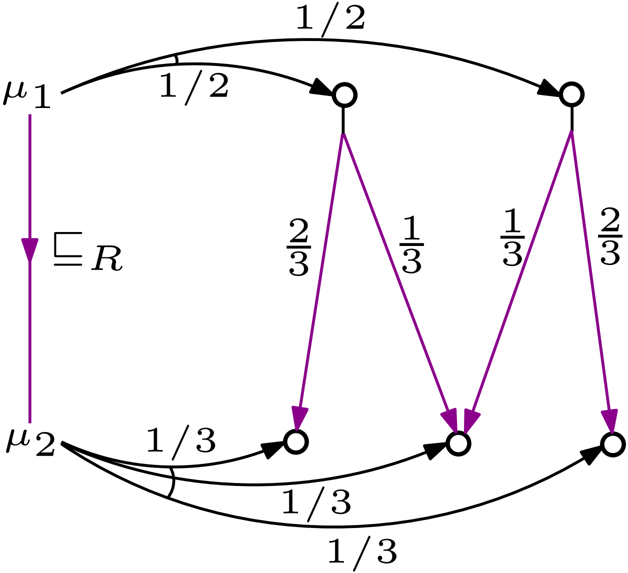
Towards being Optimal



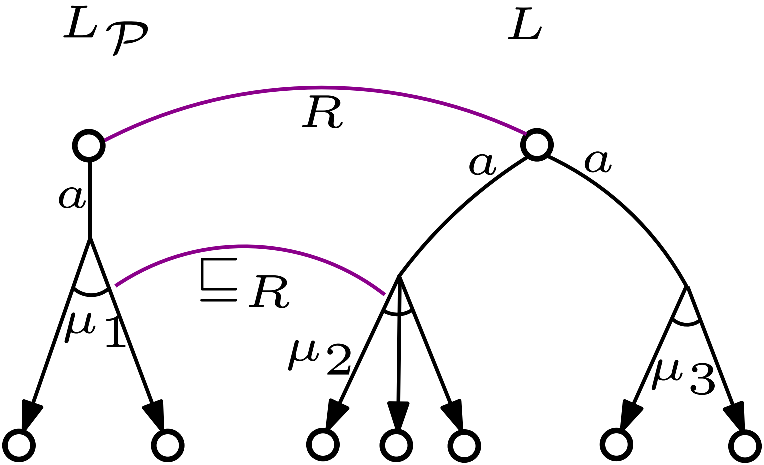
Towards being Optimal



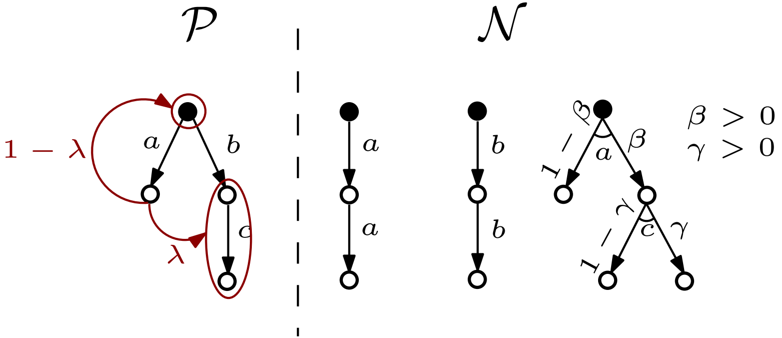
Stochastic Partition



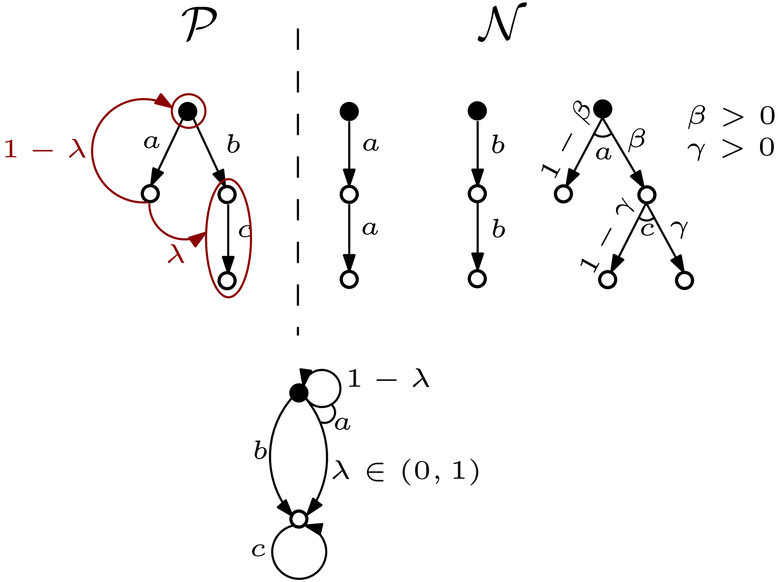
Stochastic Partition



Stochastic Partition



Stochastic Partition



Stochastic Partition

Carnegie Mellon

The Algorithm Find the *least-sized* stochastic partition Π such that $L_{\mathcal{P}}/\Pi$ is consistent!

The Algorithm Find the *least-sized* stochastic partition Π such that $L_{\mathcal{P}}/\Pi$ is consistent!
Optimal algorithm!

Is there a consistent (stochastic) partition of $L_{\mathcal{P}}$ of size at most k?

Is there a consistent (stochastic) partition of $L_{\mathcal{P}}$ of size at most k?

Reduction to Satisfiability over Linear Rational Arithmetic.

Our Goals

Question 1 How to automatically infer a H consistent with all the counterexamples? ✓

Our Goals

Question 1 How to automatically infer a H consistent with all the counterexamples? ✓

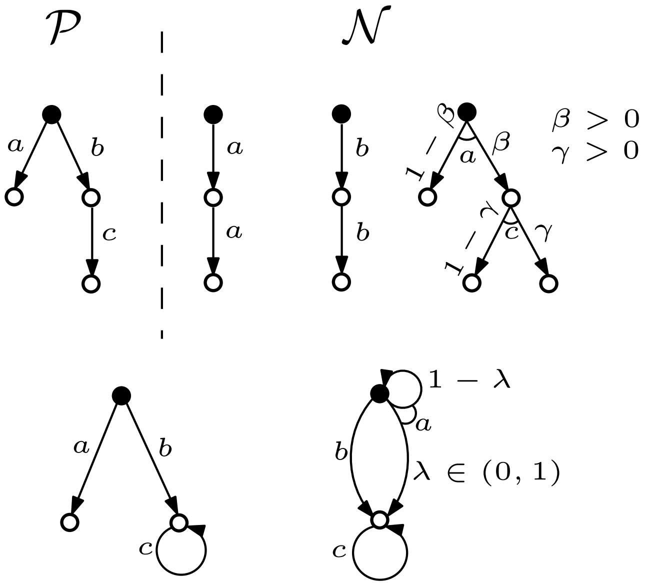
Question 2 Will the sequence of hypotheses converge?

Partition-based approach guarantees termination.

Partition-based approach guarantees termination.

If every conjecture should be of the least number of states, learning *U* is undecidable.

Learning an Unknown LPTS U



Learning an Unknown LPTS U

Carnegie Mellon

Partition-based approach guarantees termination.

When stochastic partitions are used, the sequence of hypotheses need not converge.

Stochastic Partitions give a semi-algorithm for learning an LPTS.

Our Goals

Question 1 How to automatically infer a *H* consistent with all the counterexamples? ✓

Question 2 Will the sequence of hypotheses converge? ✓

Conclusion

- Two algorithms for learning consistent LPTSes from stochastic tree samples - using Partitions and Stochastic Partitions.
- Decidability results for learning an unknown LPTS.
- Applications to Assume-Guarantee Reasoning.

Future Directions

- Alternatives for the teacher? More than tree counterexamples?

Future Directions

- Alternatives for the teacher? More than tree counterexamples?
- How useful is minimum number of states? Perhaps, one should also consider the transitions and probabilities in the size?

Future Directions

- Alternatives for the teacher? More than tree counterexamples?
- How useful is minimum number of states? Perhaps, one should also consider the transitions and probabilities in the size?
- Weak Simulation, instead of Strong Simulation.

Future Directions

- Alternatives for the teacher? More than tree counterexamples?
- How useful is minimum number of states? Perhaps, one should also consider the transitions and probabilities in the size?
- Weak Simulation, instead of Strong Simulation.
  No known algorithm for LPTSes yet to decide Weak Simulation!

Future Directions

- Alternatives for the teacher? More than tree counterexamples?
- How useful is minimum number of states? Perhaps, one should also consider the transitions and probabilities in the size?
- Weak Simulation, instead of Strong Simulation.
  No known algorithm for LPTSes yet to decide Weak Simulation!
- New applications?

Weak-safe pCTL

Preserved by Strong Simulation [Chadha and Viswanathan 2010].

$L_1 \preceq L_2$ implies
for all weak-safe ϕ, $L_2 \models \phi$ implies $L_1 \models \phi$.

Weak-safe pCTL

$$\frac{\begin{array}{c} L_2 \preceq A \\ L_1 \parallel A \models \phi \end{array}}{L_1 \parallel L_2 \models \phi}$$

Weak-safe pCTL

$$\frac{L_2 \preceq A \qquad L_1 \parallel A \models \phi}{L_1 \parallel L_2 \models \phi}$$

Probabilistic Safety
[Kwiatkowska et al. 2010]

$$\frac{\langle \top \rangle L_2 \langle A \rangle_{\geq p_A} \quad \langle A \rangle_{\geq p_A} L_1 \langle G \rangle_{\geq p_G}}{\langle \top \rangle L_1 \parallel L_2 \langle G \rangle_{\geq p_G}}$$

Probabilistic Safety
[Kwiatkowska et al. 2010]

Carnegie Mellon

$$\frac{\langle \top \rangle L_2 \langle A \rangle_{\geq p_A} \qquad \langle A \rangle_{\geq p_A} L_1 \langle G \rangle_{\geq p_G}}{\langle \top \rangle L_1 \parallel L_2 \langle G \rangle_{\geq p_G}}$$

Incomplete!

Probabilistic Safety

$$\frac{\langle \top \rangle L_2 \langle A \rangle \quad \langle A \rangle L_1 \langle G \rangle_{\geq p_G}}{\langle \top \rangle L_1 \parallel L_2 \langle G \rangle_{\geq p_G}}$$

Probabilistic Safety

$$\frac{L_2 \preceq A \qquad \langle A \rangle L_1 \langle G \rangle_{\geq p_G}}{\langle \top \rangle L_1 \parallel L_2 \langle G \rangle_{\geq p_G}}$$

Probabilistic Safety

$$\frac{L_2 \preceq A \qquad \langle A \rangle L_1 \langle G \rangle_{\geq p_G}}{\langle \top \rangle L_1 \parallel L_2 \langle G \rangle_{\geq p_G}}$$