Deciding Probabilistic Simulation between Probabilistic Pushdown Automata and Finite-State Systems

Hongfei Fu

Lehrstuhl für Informatik 2
RWTH Aachen University

October 5, 2011

Probabilistic Behavioural Equivalences

Probabilistic Behavioural Equivalences:

- are formal notions judging whether two probabilistic sytems are “equivalent”.
- are mostly extenstions from classical non-probabilistic counterparts (e.g. bisimulation by Milner and Park)
- preserve logics such as PCTL and PCTL*.
- can be used to solve state-explosion problem.

Typical Probabilistic Behavioural Equivalences

- strong (probabilistic) bisimulation (Larsen and Skou 1991, etc.)
- strong (Probabilistic) simulation (Larsen and Jonsson 1991, etc.)
- weak (probabilistic) bisimulation (Baier and Hermanns 1997, etc.)
- weak (probabilistic) simulation (Baier *et al.* 2005, etc.)

The problem We study

The complexity of deciding strong (probabilistic) simulation between probabilistic pushdown automata (pPDA) and finite probabilistic systems.

Previous Result

For (non-probabilistic) pushdown automata (PDA):

- strong bisimilarity over PDA is decidable (Sénizergues 1998, Stirling 2000).
- weak bisimilarity over PDA is undecidable (Srba 2002).
- strong/weak similarity between PDA is undecidable (Groote and Hüttel 1994).
- strong/weak bisimilarity between PDA and finite-state systems is PSPACE-complete (Kučera and Mayr 2000, 2002).
- strong/weak similarity between PDA and finite-state systems is EXPTIME-complete (Kučera and Mayr 2010).

Previous Result, Cont'd

For probabilistic pushdown automata (pPDA):

- strong (probabilistic) bisimilarity between pPDA and finite probabilistic systems lies in EXPTIME (Tomás Brázdil *et al.* 2004).

Our Contribution

Strong (probabilistic) similarity between pPDA and finite probabilistic systems is decidable, and moreover EXPTIME-complete.

Simple Probabilistic Automata (Segala and Lynch 1995)

Definition

A *simple probabilistic automata* (simple PA) is a tuple (S, A, Ω) where

- S is a countable set of *states*.
- A is a countbale set of *actions*.
- $\Omega \subseteq S \times A \times \mathcal{D}(S)$ is a set of *transitions*

where $\mathcal{D}(S)$ is the set of probability distributions over S.

A transition of Ω has the form (s, a, μ) (written as $s \xrightarrow{a} \mu$) with $s \in S$, $a \in A$ and $\mu \in \mathcal{D}(S)$.

Simple Probabilistic Automata

- Ordinary (non-probabilistic) transition systems can be seen as a simple PA where all transtions lead to a dirac distribution.
- Markov Desicion Processes (MDP) can be seen as a simple PA where the transition set Ω is a function on $S \times A$ (except for a labelling function).

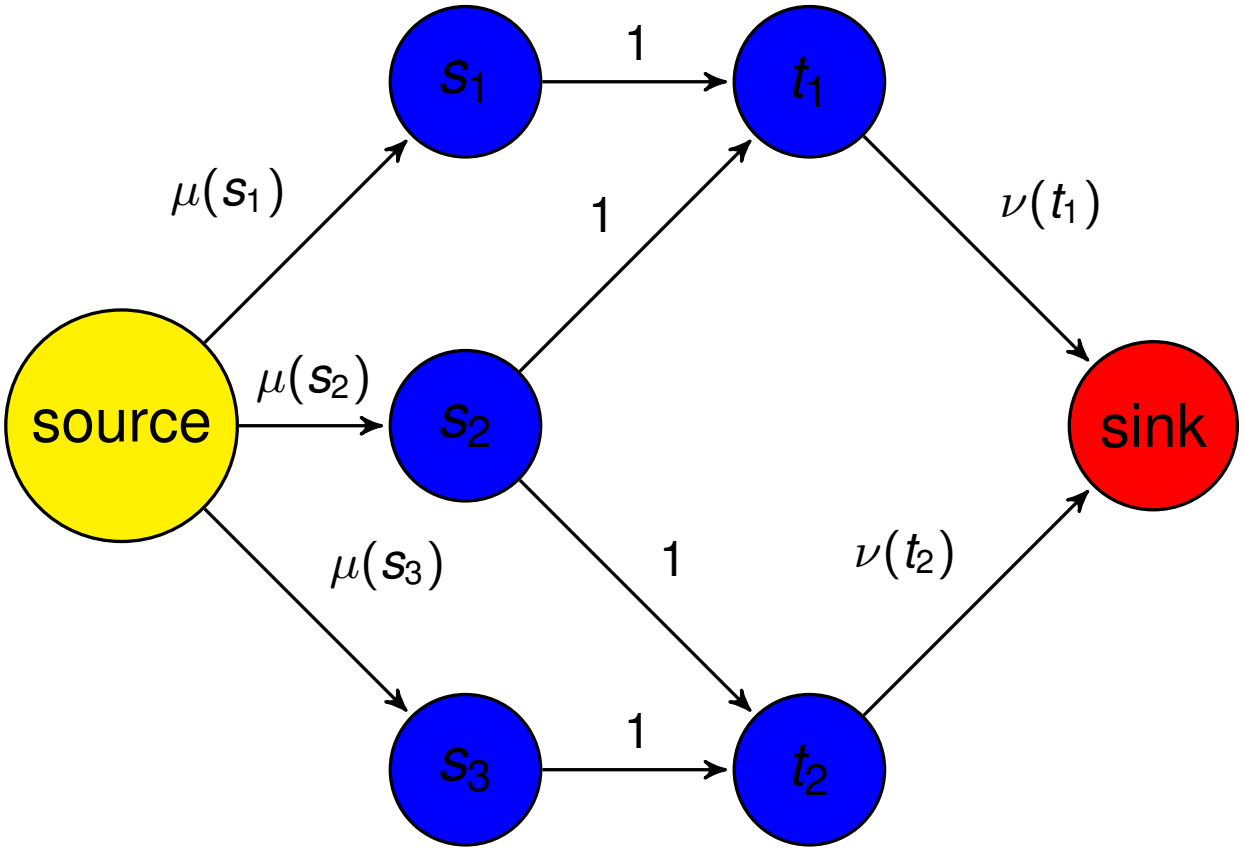
Lifting Definition (Jonsson and Larsen 1991)

Definition

Let $\mathcal{R} \subseteq S \times S$. The relation $\overline{\mathcal{R}} \subseteq \Delta(S) \times \Delta(S)$ is defined as follows: for any $(\mu, \nu) \in \Delta(S) \times \Delta(S)$, $\mu \overline{\mathcal{R}} \nu$ iff there exists a weight function $w : S \times S \to [0, 1]$ such that

- for any $s \in S$, $\sum_{t \in S} w(s, t) = \mu(s)$
- for any $t \in S$, $\sum_{s \in S} w(s, t) = \nu(t)$
- for any $(s, t) \in S \times S$, $w(s, t) > 0$ implies $(s, t) \in \mathcal{R}$

The Intuition: A Network Flow Problem



$\mu \overline{\mathcal{R}} \nu$ iff the value of the maximum network flow equals 1.
(Here $\mathcal{R} = \{(s_1, t_1), (s_2, t_1), (s_2, t_2), (s_3, t_2)\}$.)

Strong (Probabilistic) Simulation (Segala and Lynch 1995)

Definition

Let (S, A, Ω) be a simple PA. A relation $\mathcal{R} \subseteq S \times S$ is a *strong probabilistic simulation* iff for all $(s, t) \in \mathcal{R}$,

- whenever $s \xrightarrow{a} \mu$, there is $t \xrightarrow{a} \nu$ such that $\mu \overline{\mathcal{R}} \nu$;
- $Act(s) = Act(t)$.

The strong similarity $\sqsubseteq$ is the greatest strong probabilistic simulation.

The condition $Act(s) = Act(t)$

This definition:

- is a probabilistic extension of $\frac{2}{3}$-bisimulation by Larsen and Skou.
- preserves PCTL formulae without $EU_{\geq p}$ operator (Segala).

Probabilistic Pushdown Automata (pPDA)

Definition

A probabilistic pushdown automata P is a tuple $(Q, \Sigma, \Gamma, \Delta)$ where

- $Q = \{p, q \dots\}$ is a finite set of *control states*;
- $\Sigma = \{a, b \dots\}$ is a finite set of *labels*;
- $\Gamma = \{X, Y \dots\}$ is a finite set of *stack symbols*;
- $\Delta \subseteq (Q \times \Gamma) \times \Sigma \times \mathcal{D}(\mathcal{C})$ is a finite set of *transition rules*

where $\mathcal{C} = Q \times \Gamma^*$ is the set of configurations of P.

Probabilistic Pushdown Automata

- A transition rule has the form $pX \overset{a}{\rightarrowtail} \mu$ with $p \in Q$, $a \in \Sigma$, $X \in \Gamma$ and $\mu \in \mathcal{D}(\mathcal{C})$
- An extension of the fully probabilistic version by Esparza *et al.* (2004)

Probabilistic Pushdown Automata, Cont'd

Definition

A pPDA $(Q, \Sigma, \Gamma, \Delta)$ defines a simple PA (S, A, Ω) as follows:

- $S = \mathcal{C} = Q \times \Gamma^*$
- $A = \Sigma$
- $\Omega = \{pX\alpha \xrightarrow{a} \mu_\alpha \mid \alpha \in \Gamma^*, pX \overset{a}{\rightarrowtail} \mu \in \Delta\}$

where the distribution $\mu_\alpha \in \mathcal{D}(S)$ is defined by:

$$\mu_\alpha(p\beta) = \begin{cases} \mu(p\gamma) & \text{if } \beta = \gamma\alpha \\ 0 & \text{otherwise} \end{cases}$$

Problem Restatement

INPUT: a configuration $p\alpha$ under a pPDA $(Q, \Sigma, \Gamma, \Delta)$ and a state f under a finite PA (F, A, Ω)
OUTPUT: whether $p\alpha \sqsubseteq f$

Problem Restatement

INPUT: a configuration $p\alpha$ under a pPDA $(Q, \Sigma, \Gamma, \Delta)$ and a state f under a finite PA (F, A, Ω)
OUTPUT: whether $p\alpha \sqsubseteq f$

INPUT: a configuration $p\alpha$ under a pPDA $(Q, \Sigma, \Gamma, \Delta)$ and a state f under a finite PA (F, A, Ω)
OUTPUT: whether $f \sqsubseteq p\alpha$

Our Method

We follow the method of “Extended Stack Symbol” developed by Colin Stirling which is originally used to demonstrate the decidability of strong bisimilarity over (non-probabilistic) pushdown automata.

Our Mehtod

- Firstly, we introduce extended stack symbols into the original pPDA.
- Secondly, we establish a tableaux proof system for the relation $\sqsubseteq$.
- Thirdly, we perform a refinement technique to obtain the EXPTIME upperbound.

Stirling's Extended Stack Symbol

Stirling's Method:

- An extended stack symbol U is a function $Q \longrightarrow Q \times \Gamma^*$.
- The semantics goes as follows: the configuration pU behaves exactly as $U(p)$.
- Then a tableaux proof system which involves extended stack symbols is established for strong bisimilarity.
- From the tableaux proof system, the decidability result is obtained.

Extended Stack Symbols in Our Setting

- An extended stack symbol U is a function $Q \longrightarrow 2^F$ (F is the state set of the finite PA).
- The semantics is different: intuitively, $U(q)$ represents the set of states f such that "$q \sqsubseteq f$" already holds.
- We establish a tableaux proof system based on our extended stack symbols.
- We prove the EXPTIME upperbound through a refinement technique related to the tableaux proof system.

Extended Stack Symbols: The Intuition

- An extended stack symbol U is a function $Q \to 2^F$.
- Suppose $pX\alpha \sqsubseteq f$, we expect that "$pXU \sqsubseteq f$" where

$$U(q) := \{g \in F \mid q\alpha \sqsubseteq g\}, \text{ for all } q \in Q$$

Extended Stack Symbols: The Intuition

- An extended stack symbol U is a function $Q \longrightarrow 2^F$.
- Suppose $pX\alpha \sqsubseteq f$, we expect that "$pXU \sqsubseteq f$" where

$$U(q) := \{g \in F \mid q\alpha \sqsubseteq g\}, \text{ for all } q \in Q$$

- The relation $\sqsubseteq$ (which is originally defined for normal stack symbols) should be formally extended to extended stack symbols.

The extension of The Simulation Semantics

Under our semantics:

The extension of The Simulation Semantics

Under our semantics:

Theorem

If $pX\alpha \sqsubseteq f$ then $pXU \sqsubseteq f$ where $U(q) := \{g \in F \mid q\alpha \sqsubseteq g\}$, for arbitrary $q \in Q$.

The extension of The Simulation Semantics

Under our semantics:

Theorem

If $pX\alpha \sqsubseteq f$ then $pXU \sqsubseteq f$ where $U(q) := \{g \in F \mid q\alpha \sqsubseteq g\}$, for arbitrary $q \in Q$.

Theorem

If $pXU \sqsubseteq f$ and $q\alpha \sqsubseteq g$ for all $q \in Q$ and $g \in U(q)$, then $pX\alpha \sqsubseteq f$.

Our Tableaux Proof System

Two types of tableaux rules:

- Unfolding Rules
- Reduction Rules

Unfolding Rules

$$\left(\frac{\text{Goal}}{\text{Subgoals}}:\right) \qquad \frac{(pXU, f)}{(q_1\alpha_1, f_1), \ldots, (q_n\alpha_n, f_n)}$$

satisfying

- $Act(pX) = Act(f) \neq \emptyset$
- U is an extended stack symbol
- whenever $pXU \xrightarrow{a} \mu$, there is $f \xrightarrow{a} \nu$ such that $\mu \overline{\mathcal{R}} \nu$, where

$$\mathcal{R} := \{(q_1\alpha_1, f_1), \ldots, (q_n\alpha_n, f_n)\}$$

Reduction Rules

$$\left(\frac{\text{Goal}}{\text{Subgoals}} : \right) \qquad \frac{(pX\alpha, f)}{(q_1\alpha, f_1), \ldots, (q_n\alpha, f_n), (pXU, f)}$$

satisfying

- $Act(pX) = Act(f) \neq \emptyset$
- U is an extended stack symbol
- $\{(q\alpha, g) \mid q \in Q, g \in U(q)\} = \{(q_1\alpha, f_1), \ldots, (q_n\alpha, f_n)\}$

Soundness and Completeness

Theorem

$pX\alpha \sqsubseteq f$ iff there is a successful tableaux tree rooted at $(pX\alpha, f)$

Soundness and Completeness

Theorem

$pX\alpha \sqsubseteq f$ iff there is a successful tableaux tree rooted at $(pX\alpha, f)$

Theorem

The problem if $pX\alpha \sqsubseteq f$ is decidable.

Refinement Technique

Refinement Procedure for $pX\alpha \sqsubseteq f$:

- We start from a finite goal set G_0 of exponential size such that $(pX\alpha, f) \in G_0$
- We construct G_{n+1} from G_n as follows:

$$\begin{aligned} G_{n+1} = \quad & \{(q\beta, g) \in G_n \mid (q\beta, g) \text{ is a successful node or} \\ & \exists G \subseteq G_n.((q\beta, g), G) \text{ is a legitimate tableaux rule.}\} \end{aligned}$$

Refinement Technique

- Then there is $m \in \mathbb{N}$ such that $G_{m+1} = G_m$ since G_0 is finite.
- We proved that $(pX\alpha, f) \in G_m$ iff $pX\alpha \sqsubseteq f$.

EXPTIME-ness

Theorem

The problem if $pX\alpha \sqsubseteq f$ (resp. $f \sqsubseteq pX\alpha$) lies in EXPTIME.

EXPTIME-ness

Theorem

The problem if $pX\alpha \sqsubseteq f$ (resp. $f \sqsubseteq pX\alpha$) lies in EXPTIME.

Theorem

If $|Q|$ and $|F|$ is fixed, then the problem lies in PTIME.

EXPTIME-Completeness

- By a straightforward reduction from a non-probabilistic result by Kučera and Mayr, we can prove that deciding strong probabilistic simulation between pPDA and finite probabilistic system in both directions is EXPTIME-hard.
- Deciding strong probabilistic simulation between pPDA and finite probabilistic system in both directions is EXPTIME-complete.

Summary

- We have shown that deciding strong probabilistic simulation between pPDA and finite probabilistic system in both directions lies in EXPTIME.
- Then the EXPTIME-completeness follows from a prevous hardness result by Kučera and Mayr.

Summary

- Further if the number of control states and the number of states of the finite PA is fixed, then the problem is polynomial-time decidable.
- All our results can be extended to combined simulations (Segala and Lynch 1995).

Summary

- We established extended stack symbols and developed a tableaux proof system, following the method by Colin Stirling
- Further we devised a refinement technique which solves the problem in EXPTIME.