On the Memory Consumption of Probabilistic Pushdown Automata

Tomáš Brázdil\textsuperscript{1}  \quad Javier Esparza\textsuperscript{2}  \quad Stefan Kiefer\textsuperscript{3}

\begin{itemize}
\item \textsuperscript{1}Masaryk University, Brno (Czech Republic)
\item \textsuperscript{2}TU München (Germany)
\item \textsuperscript{3}University of Oxford (UK)
\end{itemize}

FSTTCS (Kanpur, India), 15 December, 2009
Probabilistic pushdown systems are a model for probabilistic procedural programs.

A run:

\[ X \xrightarrow{1/2} \epsilon \]

\[ Y \xrightarrow{2/3} X \]

Quantitative Properties of a run:
- its probability (product of the probabilities on the arrows)
- its time (number of steps till \( \epsilon \))
- its memory (longest configuration, "maximal stack height")
Probabilistic pushdown systems are a model for probabilistic procedural programs.

A run: \( X \)
Probabilistic pushdown systems are a model for probabilistic procedural programs.

A run:

\[ X \xrightarrow{1/2} YX \]

\[ X \xrightarrow{1/2} \varepsilon \]

\[ Y \xrightarrow{2/3} X \]

\[ Y \xrightarrow{1/3} \varepsilon \]

Quantitative Properties of a run:
- its probability (product of the probabilities on the arrows)
- its time (number of steps until \( \varepsilon \))
- its memory (longest configuration, "maximal stack height")
Probabilistic pushdown systems are a model for probabilistic procedural programs.

A run:

\[
\begin{align*}
X & \xleftarrow{1/2} YX & \quad & Y \xrightarrow{2/3} X \\
X & \xleftarrow{1/2} \varepsilon & \quad & Y \xleftarrow{1/3} \varepsilon \\
& \hspace{1cm} X \xrightarrow{1/2} YX \xrightarrow{2/3} XX
\end{align*}
\]
Probabilistic pushdown systems are a model for probabilistic procedural programs.

A run:

\[
\begin{align*}
X & \xrightarrow{1/2} YX & Y & \xrightarrow{2/3} X \\
X & \xrightarrow{1/2} \varepsilon & Y & \xrightarrow{1/3} \varepsilon
\end{align*}
\]

\[
\begin{align*}
X & \xrightarrow{1/2} YX & Y & \xrightarrow{2/3} X & \xrightarrow{1/2} YXX
\end{align*}
\]
Probabilistic pushdown systems are a model for probabilistic procedural programs.

A run: 

\[
X \xrightarrow{1/2} YX \quad Y \xrightarrow{2/3} X \\
X \xrightarrow{1/2} \varepsilon \quad Y \xrightarrow{1/3} \varepsilon
\]

\[
X \xrightarrow{1/2} YX \xrightarrow{2/3} XX \xrightarrow{1/2} YXX \xrightarrow{1/3} XX
\]
Probabilistic pushdown systems are a model for probabilistic procedural programs.

A run:

\[
X \xleftarrow{1/2} YX \quad Y \xrightarrow{2/3} X \\
X \xleftarrow{1/2} \varepsilon \quad Y \xrightarrow{1/3} \varepsilon
\]

\[
X \xleftarrow{1/2} YX \xrightarrow{2/3} XX \xrightarrow{1/2} YXX \xrightarrow{1/3} XX \xrightarrow{1/2} X
\]
Probabilistic pushdown systems are a model for probabilistic procedural programs.

\[
\begin{align*}
X & \xrightarrow{1/2} YX \\
X & \xrightarrow{1/2} \varepsilon \\
Y & \xrightarrow{2/3} X \\
Y & \xrightarrow{1/3} \varepsilon
\end{align*}
\]

A run:

\[
\begin{align*}
X & \xrightarrow{1/2} YX \xrightarrow{2/3} XX \xrightarrow{1/2} YXX \xrightarrow{1/3} XX \xrightarrow{1/2} X \xrightarrow{1/2} \varepsilon
\end{align*}
\]
Probabilistic pushdown systems are a model for probabilistic procedural programs.

A run:

\[
\begin{align*}
X & \xrightarrow{1/2} YX \\
X & \xrightarrow{1/2} \varepsilon
\end{align*}
\]

\[
\begin{align*}
Y & \xrightarrow{2/3} X \\
Y & \xrightarrow{1/3} \varepsilon
\end{align*}
\]

(no control states in this talk)
Probabilistic pushdown systems are a model for probabilistic procedural programs.

$$X \xrightarrow{1/2} YX \quad Y \xrightarrow{2/3} X$$

$$X \xrightarrow{1/2} \varepsilon \quad Y \xrightarrow{1/3} \varepsilon$$

A run:

$$X \xrightarrow{1/2} YX \xrightarrow{2/3} XX \xrightarrow{1/2} YXX \xrightarrow{1/3} XX \xrightarrow{1/2} X \xrightarrow{1/2} \varepsilon$$

(no control states in this talk)

Quantitative Properties of a run:

- its probability (product of the probabilities on the arrows)
- its time (number of steps till $\varepsilon$)
- its memory (longest configuration, “maximal stack height”)

Tomáš Brázdil, Javier Esparza, Stefan Kiefer

Memory Consumption of Probabilistic Pushdown Automata
Motivation

Probabilistic Pushdown Systems are a model for probabilistic programs with unrestricted recursion.

Well-studied:

- model-checking for temporal logics [Etessami, Yannakakis, Esparza, Kučera, Mayr]
- long-run behaviour [Brázdil, Esparza, Kučera]
- games [Etessami, Yannakakis]

The basic quantities time and memory are random variables.

This talk is about the memory.
Motivation

Probabilistic Pushdown Systems are a model for probabilistic programs with unrestricted recursion.

Well-studied:

- model-checking for temporal logics [Etessami, Yannakakis, Esparza, Kučera, Mayr]
- long-run behaviour [Brázdil, Esparza, Kučera]
- games [Etessami, Yannakakis]

The basic quantities time and memory are random variables.

This talk is about the memory.

What is the distribution of the memory?
What is the probability of reaching height $\geq 3$?

Idea: Construct the Markov chain with all stacks of height $\leq 2$. 
Computing the Distribution of the Memory

What is the probability of reaching height $\geq 3$?

$Idea$: Construct the Markov chain with all stacks of height $\leq 2$.

$Problem$: There may be $2^k$ states with height $\leq k$. 

$$
\begin{align*}
X & \xrightarrow{1/2} YX \\
X & \xrightarrow{1/2} \varepsilon \\
Y & \xrightarrow{2/3} X \\
Y & \xrightarrow{1/3} \varepsilon
\end{align*}
$$

Tomáš Brázdil, Javier Esparza, Stefan Kiefer
Computing the Distribution of the Memory

What is the probability of reaching height $\geq 3$?

Idea: Construct the Markov chain with all stacks of height $\leq 2$.

Problem:
There may be $2^k$ states with height $\leq k$.

Idea:
The Markov chain has a regular structure. Exploit that.
Linear Equation Systems for the Distribution

\[ X \xleftarrow{1/2} YX \quad Y \xrightarrow{2/3} X \]

\[ X \xleftarrow{1/2} \varepsilon \quad Y \xrightarrow{1/3} \varepsilon \]

Let \( p[k]_X := \) probability of reaching \( \text{height} \geq k \) if starting with \( X \).
Compute \( p[k]_X \) by solving linear equations. For instance:

\[
\begin{align*}
    p[10]_X &= 1/2 \cdot (p[9]_Y + p[10]_X) + 1/2 \cdot 0 \\
    p[10]_Y &= 2/3 \cdot p[10]_X + 1/3 \cdot 0
\end{align*}
\]
Let $p[k]_X :=$ probability of reaching height $\geq k$ if starting with $X$. Compute $p[k]_X$ by solving linear equations. For instance:

\[
\begin{align*}
  p[10]_X &= \frac{1}{2} \cdot (p[9]_Y + t[9]_Y \cdot p[10]_X) + \frac{1}{2} \cdot 0 \\
  p[10]_Y &= \frac{2}{3} \cdot p[10]_X + \frac{1}{3} \cdot 0
\end{align*}
\]

where $t[k]_Y$ is the probability of terminating and having height $< k$. 

\[
\begin{align*}
  X &\xleftarrow{1/2} YX \\
  X &\xleftarrow{1/2} \varepsilon \\
  Y &\xleftarrow{2/3} X \\
  Y &\xleftarrow{1/3} \varepsilon
\end{align*}
\]
Linear Equation Systems for the Distribution

\[
\begin{align*}
X & \xrightarrow{1/2} YX \\
X & \xrightarrow{1/2} \varepsilon \\
Y & \xrightarrow{2/3} X \\
Y & \xrightarrow{1/3} \varepsilon
\end{align*}
\]

Let \( p[k]_X \) := probability of reaching height \( \geq k \) if starting with \( X \). Compute \( p[k]_X \) by solving linear equations. For instance:

\[
\begin{align*}
p[10]_X &= 1/2 \cdot (p[9]_Y + t[9]_Y \cdot p[10]_X) + 1/2 \cdot 0 \\
p[10]_Y &= 2/3 \cdot p[10]_X + 1/3 \cdot 0
\end{align*}
\]

where \( t[k]_Y \) is the probability of terminating and having height \( < k \).

\[
\begin{align*}
t[10]_X &= 1/2 \cdot t[9]_Y \cdot t[10]_X + 1/2 \cdot 1 \\
t[10]_Y &= 2/3 \cdot t[10]_X + 1/3 \cdot 1
\end{align*}
\]
Proposition

The vector \( p[k] \) can be computed by setting up and solving linear equation systems. It can be done in \( \mathcal{O}(k \cdot |\Gamma|^3) \) arithmetic operations.
The vector $p[k]$ can be computed by setting up and solving linear equation systems. It can be done in $O(k \cdot |\Gamma|^3)$ arithmetic operations.

Is this fast enough?
Proposition

The vector $p[k]$ can be computed by setting up and solving linear equation systems. It can be done in $O(k \cdot |\Gamma|^3)$ arithmetic operations.

Is this fast enough? Depends.
Proposition

The vector $p[k]$ can be computed by setting up and solving linear equation systems. It can be done in $O(k \cdot |\Gamma|^3)$ arithmetic operations.

Is this fast enough? Depends.
What about the asymptotics for large $k$?
Proposition

The vector $p[k]$ can be computed by setting up and solving linear equation systems. It can be done in $O(k \cdot |\Gamma|^3)$ arithmetic operations.

Is this fast enough? Depends.
What about the asymptotics for large $k$?
What about the expectation of the memory consumption?
The vector $p[k]$ can be computed by setting up and solving linear equation systems. It can be done in $O(k \cdot |\Gamma|^3)$ arithmetic operations.

Is this fast enough? Depends.
What about the asymptotics for large $k$?
What about the expectation of the memory consumption?

In the following we assume finite expectation of the memory (which is the most important case).
After some (non-trivial but non-interesting) normalizations:

There is a (nonnegative) matrix $A(x) \in \mathbb{R}^{\Gamma \times \Gamma}$ that depends on a (nonnegative) vector $x \in \mathbb{R}^{\Gamma}$ such that:

- $A(x)$ increases monotonically with $x$,
- $p[k + 1] = A(t[k]) \cdot p[k]$
After some (non-trivial but non-interesting) normalizations:

There is a (nonnegative) matrix $A(x) \in \mathbb{R}^{\Gamma \times \Gamma}$ that depends on a (nonnegative) vector $x \in \mathbb{R}^{\Gamma}$ such that:

- $A(x)$ increases monotonically with $x$,
- $p[k + 1] = A(t[k]) \cdot p[k]$

The sequence $t[1] \leq t[2] \leq \ldots$ converges to a limit $t$, where $t$ is the vector of termination probabilities.

By monotonicity and continuity, the sequence $A(t[1]) \leq A(t[2]) \leq \ldots$ also converges to a limit $A(t) = A$. 
Putting the Equations in Matrix Form

After some (non-trivial but non-interesting) normalizations:

There is a (nonnegative) matrix $A(x) \in \mathbb{R}^{\Gamma \times \Gamma}$ that depends on a (nonnegative) vector $x \in \mathbb{R}^{\Gamma}$ such that:

- $A(x)$ increases monotonically with $x$,
- $p[k + 1] = A(t[k]) \cdot p[k]$

The sequence $t[1] \leq t[2] \leq \ldots$ converges to a limit $t$, where $t$ is the vector of termination probabilities.

By monotonicity and continuity, the sequence $A(t[1]) \leq A(t[2]) \leq \ldots$ also converges to a limit $A(t) = A$.

It follows:

$$p[k + 1] = A(t[k]) \cdot p[k] \leq A \cdot p[k] \leq A^k \cdot p[1] = A^k \cdot 1$$
On the previous slide: $p[k] \leq A^{k-1} \cdot 1$, so $A^{k-1} \cdot 1$ is an upper bound on $p[k]$.

Advantage: compute $A^{k-1}$ by repeated squaring: $A, A^2, A^4, \ldots$  
⇒ only $O(\log k \cdot |\Gamma|^3)$ operations  
⇒ safe overapproximation for large $k$

How tight is the overapproximation?
Approximating the Distribution

On the previous slide: \( p[k] \leq A^{k-1} \cdot 1 \), so \( A^{k-1} \cdot 1 \) is an upper bound on \( p[k] \).

Advantage: compute \( A^{k-1} \) by repeated squaring: \( A, A^2, A^4, \ldots \)
\( \Rightarrow \) only \( \mathcal{O}(\log k \cdot |\Gamma|^3) \) operations
\( \Rightarrow \) safe overapproximation for large \( k \)

How tight is the overapproximation?

**Proposition**

There is a real number \( d \) with \( 0 < d \leq 1 \) such that for all \( k \):

\[
d \cdot A^{k-1} \cdot 1 \leq p[k] \leq A^{k-1} \cdot 1.
\]
Proposition

There is a real number $d$ with $0 < d \leq 1$ such that for all $k$:

$$d \cdot A^{k-1} \cdot 1 \leq p[k] \leq A^{k-1} \cdot 1.$$  

So, $p[k]$ essentially depends on the spectral radius $\rho$ of $A$.  

On the previous slide:
Approximating the Distribution

On the previous slide:

**Proposition**

There is a real number $d$ with $0 < d \leq 1$ such that for all $k$:

$$d \cdot A^{k-1} \cdot 1 \leq p[k] \leq A^{k-1} \cdot 1.$$ 

So, $p[k]$ essentially depends on the spectral radius $\rho$ of $A$.

**Theorem**

We have $\rho < 1$. The proposition above implies:

$$p[k] \in \Theta(\rho^k)$$
What is the expectation of the memory?

This expectation $EM$ equals $\sum_{k=1}^{\infty} p[k]x$. 
What is the **expectation** of the memory?

This expectation $EM$ equals $\sum_{k=1}^{\infty} p[k] x$.

$EM$ can be (under-)approximated by $UM[\ell] := \sum_{k=1}^{\ell} p[k] x$.

**Theorem**

The sequence $(UM[\ell])_\ell$ converges to $EM$. More precisely, one can compute $a > 0$ and $0 < b < 1$ with

$$EM - UM[\ell] \leq a \cdot b^\ell,$$

so the error decays exponentially.
Most presented results hold if the expectation is finite.
How to decide whether the expectation is finite?

- $X \xrightarrow{2/3} XX$, $X \xrightarrow{1/3} \varepsilon$: $P(M = \infty) = 1/2 > 0$, so $EM = \infty$

- $X \xrightarrow{1/2} XX$, $X \xrightarrow{1/2} \varepsilon$: $P(M = \infty) = 0$, but still $EM \approx 1.61$
Most presented results hold if the expectation is finite. How to decide whether the expectation is finite?

- $X \xrightarrow{2/3} XX, X \xrightarrow{1/3} \varepsilon$: $P(M = \infty) = 1/2 > 0$, so $EM = \infty$

- $X \xleftarrow{1/2} XX, X \xrightarrow{1/2} \varepsilon$: $P(M = \infty) = 0$, but still $EM = \infty$
Finiteness of the Expectation

Most presented results hold if the expectation is finite. How to decide whether the expectation is finite?

1. $X \xrightarrow{2/3} XX$, $X \xrightarrow{1/3} \varepsilon$: $P(M = \infty) = 1/2 > 0$, so $EM = \infty$

2. $X \xrightarrow{1/2} XX$, $X \xrightarrow{1/2} \varepsilon$: $P(M = \infty) = 0$, but still $EM = \infty$

3. $X \xrightarrow{1/3} XX$, $X \xrightarrow{2/3} \varepsilon$: $P(M = \infty) = 0$, and $EM \approx 1.61$
Most presented results hold if the expectation is finite. How to decide whether the expectation is finite?

- $X \xrightarrow{2/3} XX, \ X \xrightarrow{1/3} \varepsilon$: $P(M = \infty) = 1/2 > 0$, so $EM = \infty$
- $X \xrightarrow{1/2} XX, \ X \xrightarrow{1/2} \varepsilon$: $P(M = \infty) = 0$, but still $EM = \infty$
- $X \xrightarrow{1/3} XX, \ X \xrightarrow{2/3} \varepsilon$: $P(M = \infty) = 0$, and $EM \approx 1.61$

**Theorem**

_Whether EM is finite can be decided_
- _in polynomial time for pushdown systems without states_
- _in polynomial space for general pushdown systems._

_The problem is PosSLP-hard (therefore unlikely in P) for general pushdown systems._
Probabilistic pushdown systems model recursive probabilistic programs.

We studied one basic random variable, the memory.

Its distribution can be computed
  
  naively (using an exponential-sized Markov chain),
  by solving linear equation systems,
  by efficiently computing overapproximations of the memory.

The expectation can be approximated, and the error decays exponentially.

Whether the expectation is finite can be decided in polynomial time for pushdown systems without states.

Open question: How to decide whether the expectation exceeds a given bound?
Thank you!