Certificates for Probabilistic Pushdown Automata via Optimistic Value Iteration

Tobias Winkler & Joost-Pieter Katoen

April 27, 2023 — TACAS 2023
Probabilistic Model Checking

Probabilistic Model
- Markov chain
- MDP
- Probabilistic TA
- ...

Property
- Reachability
- Safety
- LTL
- ...

2/17
Probabilistic Model Checking

Probabilistic Model
- Markov chain
- MDP
- Probabilistic TA
- ...

Property
- Reachability
- Safety
- LTL
- ...

[Diagram with arrows and icons]
Probabilistic Model Checking

Probabilistic Model
- Markov chain
- MDP
- Probabilistic TA
- ...

Property
- Reachability
- Safety
- LTL
- ...

Diagram: Gear with bugs and check marks.
Bugs in Model Checkers? Two solutions
Bugs in Model Checkers? Two solutions

(1) Fully formally verified model checkers
Bugs in Model Checkers? Two solutions

(1) Fully formally verified model checkers

(2) Certifying model checking algorithms: compute result + easy-to-check witness
Bugs in Model Checkers? Two solutions

(1) Fully formally verified model checkers

(2) **Certifying** model checking algorithms: compute result + *easy-to-check* witness
Bugs in Model Checkers? Two solutions

(1) Fully formally verified model checkers

(2) **Certifying** model checking algorithms: compute result + easy-to-check witness
Bugs in Model Checkers? Two solutions

(1) Fully formally verified model checkers

(2) Certifying model checking algorithms: compute result + easy-to-check witness
Bugs in Model Checkers? Two solutions

(1) Fully formally verified model checkers

(2) Certifying model checking algorithms: compute result + easy-to-check witness

This talk:

Certifying Algorithms for Probabilistic Pushdown Automata (pPDA)
Example: Random And-Or Trees

[Brázdil et al. ‘15]
Example: Random And-Or Trees

1) Every node has either 0 or 2 children, both with probability $1/2$
Example: Random And-Or Trees

1) Every node has either 0 or 2 children, both with probability 1/2

2) Leaves have value 0 or 1, again with probability 1/2 each
Example: Random And-Or Trees [Brázdil et al. ‘15]

1) Every node has either 0 or 2 children, both with probability 1/2

2) Leaves have value 0 or 1, again with probability 1/2 each

3) And/Or-nodes alternate from root to leaves
Example: Random And-Or Trees

1) Every node has either 0 or 2 children, both with probability 1/2

2) Leaves have value 0 or 1, again with probability 1/2 each

3) And/Or-nodes alternate from root to leaves

4) Root is an And-node or a leaf
Example: Random And-Or Trees

1) Every node has either 0 or 2 children, both with probability 1/2

2) Leaves have value 0 or 1, again with probability 1/2 each

3) And/Or-nodes alternate from root to leaves

4) Root is an And-node or a leaf

What is the probability that a random tree evaluates to true?
Example cont.
Example cont.

- Model tree generation/evaluation as recursive probabilistic program
• Model tree generation/evaluation as recursive probabilistic program

```c
bool and() { // main function
    prob {
        1/2: return // leaf
        (1/2: true | 1/2: false);
        1/2: { // inner node
            if(!or()) return false;
            else return or(); } } }

bool or() { // main function
    prob {
        1/2: return
        (1/2: true | 1/2: false);
        1/2: {
            if(and()) return true;
            else return and(); } } }
```
Example cont.

- Model tree generation/evaluation as recursive probabilistic program

- Use our tool PRAY to construct a pPDA

```c++
bool and() { // main function
    prob {
        1/2: return // leaf
            (1/2: true | 1/2: false);
        1/2: { // inner node
            if(!or()) return false;
            else return or(); } } }

bool or() { // main function
    prob {
        1/2: return
            (1/2: true | 1/2: false);
        1/2: {
            if(and()) return true;
            else return and(); } } }
```
Example cont.

- Model tree generation/evaluation as recursive probabilistic program
- Use our tool PRAY to construct a pPDA

bool and() { // main function
    prob {
        1/2: return // leaf
        (1/2: true | 1/2: false);
    } 1/2: { // inner node
        if(!or()) return false;
        else return or(); } } }
Example cont.

- Model tree generation/evaluation as recursive probabilistic program

- Use our tool PRAY to construct a pPDA

- Compute result:

\[
Pr(V = 0) \leq \frac{391}{933} \approx 0.42 \quad Pr(V = 1) \leq \frac{382}{657} \approx 0.58
\]

```
bool and() {  // main function
    prob {
        1/2: return // leaf
        (1/2: true | 1/2: false);
        1/2: { // inner node
            if(!or()) return false;
            else return or(); } } } } 

bool or() { 
    prob {
        1/2: return
        (1/2: true | 1/2: false);
        1/2: {
            if(and()) return true;
            else return and(); } } } }
```
Example cont.

- Model tree generation/evaluation as recursive probabilistic program
- Use our tool PRAY to construct a pPDA
- Compute result:

\[
Pr(V = 0) \leq \frac{391}{933} \approx 0.42 \quad Pr(V = 1) \leq \frac{382}{657} \approx 0.58
\]

- Correctness of result can be easily checked independently $\rightarrow$ certificate!
Probabilistic Pushdown Automata

pPDA

[Esparza et al. '04]
Probabilistic Pushdown Automata [Esparza et al. ’04]

pPDA

\[ (1/4, Z, ZZ), \quad (1, Z, \epsilon) \]

\[ (1/2, Z, \epsilon) \]
Probabilistic Pushdown Automata [Esparza et al. ’04]
pPDA

\[(q_0, (1/2, Z, \varepsilon), Z) \rightarrow (q_1, (1/4, Z, Z), \varepsilon) \]

\[(q_1, (1, Z, \varepsilon), \varepsilon) \rightarrow (q_0, (1/4, Z, Z), Z) \]

\[(q_0, (1/2, Z, \varepsilon), \varepsilon) \rightarrow (q_1, (1/4, Z, Z), \varepsilon) \]

\[(q_1, (1/4, Z, Z), \varepsilon) \rightarrow (q_0, (1/2, Z, \varepsilon), Z) \]

\[(q_0, (1, Z, \varepsilon), \varepsilon) \rightarrow (q_1, (1, Z, Z), \varepsilon) \]
Probabilistic Pushdown Automata

pPDA

[Esparza et al. ’04]
pPDA $\rightarrow$ Polynomial Equations

$\frac{1}{4}, Z, ZZ \rightarrow q_0 \rightarrow (1, Z, \epsilon) \rightarrow q_1 \rightarrow (1/2, Z, \epsilon) \rightarrow q_0$

$\frac{1}{4}, Z, \epsilon \rightarrow q_0 \rightarrow (1/4, Z, ZZ) \rightarrow q_1 \rightarrow (1, Z, \epsilon) \rightarrow q_0$

[Esparza et al. ’04]
pPDA ➔ Polynomial Equations

\[ [q_0Zq_i] = Pr(\text{Reachability probability}) \]
pPDA → Polynomial Equations

\[ [q_0Zq_i] = Pr(\begin{array}{c}
q_0 \\
Z \\
q_i
\end{array}) \]

Reachability probability

[Esparza et al. '04]
pPDA $\rightarrow$ Polynomial Equations

(1/4, Z, ZZ) $\rightarrow$ (1, Z, $\varepsilon$) $\rightarrow$ (1/2, Z, $\varepsilon$)

$q_0 \rightarrow q_1$

$[q_0Zq_i] = Pr($

Reachability probability

(I) $[q_0Zq_0] = \frac{1}{2} + \frac{1}{4}[q_0Zq_0]^2$

(II) $[q_0Zq_1] = \frac{1}{4} + \frac{1}{4}[q_0Zq_0][q_0Zq_1] + \frac{1}{4}[q_0Zq_1]$
pPDA → Polynomial Equations

\[ q_0 \xrightarrow{(1/4, Z, ZZ)} q_1 \xrightarrow{(1/4, Z, \varepsilon)} (1, Z, \varepsilon) \]

\[ [q_0 Z q_i] = Pr(Z) \]
pPDA \rightarrow \text{Polynomial Equations}

$$[q_0Zq_i] = Pr(\text{Reachability probability})$$

$$x = \frac{1}{2} + \frac{1}{4}x^2$$

$$y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y$$

[Esparza et al. '04]
Polynomial Equations

(I) \[ x = \frac{1}{2} + \frac{1}{4}x^2 \]

(II) \[ y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y \]
Polynomial Equations

(I) \( x = \frac{1}{2} + \frac{1}{4}x^2 \)

(II) \( y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y \)

• Possibly many solutions \(\rightarrow\) want the least solution \(\geq 0\)
Polynomial Equations

(Ⅰ) \[ x = \frac{1}{2} + \frac{1}{4}x^2 \]

(Ⅱ) \[ y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y \]

• Possibly many solutions → want the least solution \( \geq 0 \)
• Here: \( x = 2 - \sqrt{2} \quad y = \sqrt{2} - 1 \)
Polynomial Equations

(Ⅰ) \( x = \frac{1}{2} + \frac{1}{4}x^2 \)

(Ⅱ) \( y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y \)

• Possibly many solutions \( \rightarrow \) want the least solution \( \geq 0 \)
• Here: \( x = 2 - \sqrt{2} \quad y = \sqrt{2} - 1 \)
• Approximate solution numerically [Etessami & Yannakakis ’05]
Polynomial Equations

\[\begin{align*}
(\text{I}) \quad x &= \frac{1}{2} + \frac{1}{4}x^2 \\
(\text{II}) \quad y &= \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y
\end{align*}\]

- Possibly many solutions $\rightarrow$ want the least solution $\geq 0$
- Here: $x = 2 - \sqrt{2} \approx 0.588$  
  $y = \sqrt{2} - 1 \approx 0.414$
- Approximate solution numerically [Etessami & Yannakakis ’05]
Polynomial Equations

Possibly many solutions → want the least solution \( \geq 0 \)

Here:

\[
(\text{I}) \quad x = \frac{1}{2} + \frac{1}{4}x^2
\]

\[
(\text{II}) \quad y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y
\]

Approximate solution numerically [Etessami & Yannakakis ’05]

Problem: How to certify that approximation is “correct”?
Naive Idea to Check Solution

• Given approximation $x = 0.588$, $y = 0.414$ check

(I) $x \approx \frac{1}{2} + \frac{1}{4}x^2$

(II) $y \approx \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y$
Naive Idea to Check Solution

- Given approximation $x = 0.588$, $y = 0.414$ check

$$
\begin{align*}
(I) \quad x & \approx \frac{1}{2} + \frac{1}{4}x^2 \\
(II) \quad y & \approx \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y
\end{align*}
$$

- This is **unsound**! Doesn’t prove anything.
Certifying Solutions

• Our idea: Compute approximate solution with a special property:
Certifying Solutions

• Our idea: Compute approximate solution with a special property:

\[
\begin{align*}
(\text{i}) \quad & x \geq \frac{1}{2} + \frac{1}{4}x^2 \\
(\text{ii}) \quad & y \geq \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y
\end{align*}
\]
Certifying Solutions

• Our idea: Compute approximate solution with a special property:

\[(I) \quad x \geq \frac{1}{2} + \frac{1}{4}x^2\]

\[(II) \quad y \geq \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y\]

• Then: \((x, y) \geq \) (least solution)
Certifying Solutions

• Our idea: Compute approximate solution with a special property:

\[(l) \quad x \geq \frac{1}{2} + \frac{1}{4}x^2\]

\[(ll) \quad y \geq \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y\]

• Then: \((x,y) \geq \) (least solution)

• \((x,y)\) is a self-certifying upper bound!
Certifying Solutions

- Our idea: Compute approximate solution with a special property:

  \[(I) \quad x \geq \frac{1}{2} + \frac{1}{4}x^2\]
  \[(II) \quad y \geq \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y\]

- Then: \((x, y) \geq \text{(least solution)}\)

- \((x, y)\) is a self-certifying upper bound!
Computing Certifying Solutions

“Optimistic” Value Iteration

(I) \[ x = \frac{1}{2} + \frac{1}{4}x^2 \]

(II) \[ y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y \]
Computing Certifying Solutions

“Optimistic” Value Iteration

\( (I) \quad x = \frac{1}{2} + \frac{1}{4}x^2 \)

\( (II) \quad y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y \)
Computing Certifying Solutions

“Optimistic” Value Iteration

(I) \[ x = \frac{1}{2} + \frac{1}{4}x^2 \]

(II) \[ y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y \]
Computing Certifying Solutions
“Optimistic” Value Iteration

(I) \[ x = \frac{1}{2} + \frac{1}{4}x^2 \]

(II) \[ y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y \]
Computing Certifying Solutions

“Optimistic” Value Iteration

(I) \[ x = \frac{1}{2} + \frac{1}{4}x^2 \]

(II) \[ y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y \]
Computing Certifying Solutions
“Optimistic” Value Iteration

(Ⅰ) \[ x = \frac{1}{2} + \frac{1}{4}x^2 \]

(Ⅱ) \[ y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y \]
Computing Certifying Solutions

“Optimistic” Value Iteration

(I) \[ x = \frac{1}{2} + \frac{1}{4}x^2 \]

(II) \[ y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y \]
Computing Certifying Solutions

“Optimistic” Value Iteration

(I) \( x = \frac{1}{2} + \frac{1}{4}x^2 \)

(II) \( y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y \)
Computing Certifying Solutions

“Optimistic” Value Iteration

(I) \[ x = \frac{1}{2} + \frac{1}{4}x^2 \]

(II) \[ y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y \]
Computing Certifying Solutions

“Optimistic” Value Iteration

(I) \[ x = \frac{1}{2} + \frac{1}{4}x^2 \]

(II) \[ y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y \]
Computing Certifying Solutions
“Optimistic” Value Iteration

(I) \[ x = \frac{1}{2} + \frac{1}{4}x^2 \]

(II) \[ y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y \]
Computing Certifying Solutions

“Optimistic” Value Iteration

(I) \[ x = \frac{1}{2} + \frac{1}{4}x^2 \]

(II) \[ y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y \]
Computing Certifying Solutions
“Optimistic” Value Iteration

(I) \[ x = \frac{1}{2} + \frac{1}{4}x^2 \]

(II) \[ y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y \]
Computing Certifying Solutions

“Optimistic” Value Iteration

(I) \[ x = \frac{1}{2} + \frac{1}{4} x^2 \]

(II) \[ y = \frac{1}{4} + \frac{1}{4} xy + \frac{1}{4} y \]
Computing Certifying Solutions

“Optimistic” Value Iteration

(I) \[ x = \frac{1}{2} + \frac{1}{4} x^2 \]

(II) \[ y = \frac{1}{4} + \frac{1}{4} xy + \frac{1}{4} y \]
Computing Certifying Solutions

“Optimistic” Value Iteration

(I) \[ x = \frac{1}{2} + \frac{1}{4}x^2 \]

(II) \[ y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y \]
Computing Certifying Solutions

“Optimistic” Value Iteration

(I) \[ x = \frac{1}{2} + \frac{1}{4}x^2 \]

(II) \[ y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y \]
Computing Certifying Solutions

“Optimistic” Value Iteration

(I) \( x = \frac{1}{2} + \frac{1}{4}x^2 \)

(II) \( y = \frac{1}{4} + \frac{1}{4}xy + \frac{1}{4}y \)

Guess optimistically, then check [Hartmanns & Kaminski ’20]
Termination?

- Algorithm does not terminate if we guess in the wrong direction
Termination?

- Algorithm does not terminate if we guess in the wrong direction

**Theorem**
Convergence is guaranteed* if guessing direction is approximately an eigenvector of the system’s Jacobi matrix evaluated at the current under-approximation.
Float vs Exact Arithmetic
Float vs Exact Arithmetic

- Certificate should be formal proof → prefer exact rational numbers
Float vs Exact Arithmetic

- Certificate should be formal proof → prefer exact rational numbers

1. Run algorithm with floats
Float vs Exact Arithmetic

- Certificate should be formal proof → prefer exact rational numbers

1. Run algorithm with floats
2. Convert result to rationals
Float vs Exact Arithmetic

- Certificate should be formal proof
  → prefer exact rational numbers

1. Run algorithm with floats
2. Convert result to rationals
3. Check $\geq$ with exact arithmetic (often fails!)
Float vs Exact Arithmetic

• Certificate should be formal proof → prefer exact rational numbers

1. Run algorithm with floats
2. Convert result to rationals
3. Check \( \geq \) with exact arithmetic (often fails!)
Application
Consistency of stochastic CFG

\[
X \to a \mid XYY \\
Y \to b \mid X \mid YY
\]
Application
Consistency of stochastic CFG

\[ X \rightarrow a \mid XYY \]
\[ Y \rightarrow b \mid X \mid YY \]

- **Consistency:** Is \( \sum_{w \in \{a,b\}^*} Pr(w) = 1 \)?
Application

Consistency of stochastic CFG

\[ X \rightarrow a \mid XYY \]
\[ Y \rightarrow b \mid X \mid YY \]

- Consistency: Is \( \sum_{w \in \{a,b\}^*} Pr(w) = 1 \)?

\[ (I) \ x = \frac{1}{2} (1 + xy^2) \]
\[ (II) \ y = \frac{1}{3} (1 + x + y^2) \]
Application
Consistency of stochastic CFG

\[ X \rightarrow a \mid XYY \]
\[ Y \rightarrow b \mid X \mid YY \]

- **Consistency:** Is \[ \sum_{w \in \{a,b\}^*} Pr(w) = 1 \]?

- **Grammar consistent \iff least solution** \((x, y) = (1, 1)\)

\[ (I) \quad x = \frac{1}{2}(1 + xy^2) \]
\[ (II) \quad y = \frac{1}{3}(1 + x + y^2) \]
Application

Consistency of stochastic CFG

\[
X \rightarrow a \mid XYY
\]

\[
Y \rightarrow b \mid X \mid YY
\]

- Consistency: Is \( \sum_{w \in \{a,b\}^*} Pr(w) = 1 ? \)

- Grammar consistent \( \iff \) least solution \((x, y) = (1, 1)\)

- Our algorithm finds certificates for inconsistency

\[
(I) \quad x = \frac{1}{2}(1 + xy^2)
\]

\[
(II) \quad y = \frac{1}{3}(1 + x + y^2)
\]
## Stochastic Grammars Benchmark

Certificates for inconsistency

<table>
<thead>
<tr>
<th>name</th>
<th>non-terminals</th>
<th>rules</th>
<th>time OVI</th>
<th>time SMT (z3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>brown</td>
<td>37</td>
<td>22,866</td>
<td>3.2s</td>
<td>TO</td>
</tr>
<tr>
<td>lemonde</td>
<td>121</td>
<td>32,885</td>
<td>40.1s</td>
<td>TO</td>
</tr>
<tr>
<td>negra</td>
<td>256</td>
<td>29,297</td>
<td>10.2s</td>
<td>37.2s</td>
</tr>
<tr>
<td>swbd</td>
<td>309</td>
<td>47,578</td>
<td>19.0s</td>
<td>TO</td>
</tr>
<tr>
<td>tiger</td>
<td>318</td>
<td>52,184</td>
<td>94.5s</td>
<td>17.5s</td>
</tr>
<tr>
<td>tuebadz</td>
<td>196</td>
<td>8,932</td>
<td>2.6s</td>
<td>15.3s</td>
</tr>
<tr>
<td>wsj</td>
<td>240</td>
<td>31,170</td>
<td>30.3s</td>
<td>TO</td>
</tr>
</tbody>
</table>
# Stochastic Grammars Benchmark

Certificates for inconsistency

<table>
<thead>
<tr>
<th>name</th>
<th>non-terminals</th>
<th>rules</th>
<th>time OVI</th>
<th>time SMT (z3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>brown</td>
<td>37</td>
<td>22,866</td>
<td>3.2s</td>
<td>TO</td>
</tr>
<tr>
<td>lemonde</td>
<td>121</td>
<td>32,885</td>
<td>40.1s</td>
<td>TO</td>
</tr>
<tr>
<td>negra</td>
<td>256</td>
<td>29,297</td>
<td>10.2s</td>
<td>37.2s</td>
</tr>
<tr>
<td>swbd</td>
<td>309</td>
<td>47,578</td>
<td>19.0s</td>
<td>TO</td>
</tr>
<tr>
<td>tiger</td>
<td>318</td>
<td>52,184</td>
<td>94.5s</td>
<td>17.5s</td>
</tr>
<tr>
<td>tuebadz</td>
<td>196</td>
<td>8,932</td>
<td>2.6s</td>
<td>15.3s</td>
</tr>
<tr>
<td>wsj</td>
<td>240</td>
<td>31,170</td>
<td>30.3s</td>
<td>TO</td>
</tr>
</tbody>
</table>

QF\_NRA aka ETR
# Stochastic Grammars Benchmark

## Certificates for inconsistency

<table>
<thead>
<tr>
<th>name</th>
<th>non-terminals</th>
<th>rules</th>
<th>time OVI</th>
<th>time SMT (z3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>brown</td>
<td>37</td>
<td>22,866</td>
<td>3.2s</td>
<td>TO</td>
</tr>
<tr>
<td>lemonde</td>
<td>121</td>
<td>32,885</td>
<td>40.1s</td>
<td>TO</td>
</tr>
<tr>
<td>negra</td>
<td>256</td>
<td>29,297</td>
<td>10.2s</td>
<td>37.2s</td>
</tr>
<tr>
<td>swbd</td>
<td>309</td>
<td>47,578</td>
<td>19.0s</td>
<td>TO</td>
</tr>
<tr>
<td>tiger</td>
<td>318</td>
<td>52,184</td>
<td>94.5s</td>
<td>17.5s</td>
</tr>
<tr>
<td>tuebadz</td>
<td>196</td>
<td>8,932</td>
<td>2.6s</td>
<td>15.3s</td>
</tr>
<tr>
<td>wsj</td>
<td>240</td>
<td>31,170</td>
<td>30.3s</td>
<td>TO</td>
</tr>
</tbody>
</table>
## Stochastic Grammars Benchmark

### Certificates for inconsistency

<table>
<thead>
<tr>
<th>name</th>
<th>non-terminals</th>
<th>rules</th>
<th>time OVI</th>
<th>time SMT (z3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>brown</td>
<td>37</td>
<td>22,866</td>
<td>3.2s</td>
<td>TO</td>
</tr>
<tr>
<td>lemonde</td>
<td>121</td>
<td>32,885</td>
<td>40.1s</td>
<td>TO</td>
</tr>
<tr>
<td>negra</td>
<td>256</td>
<td>29,297</td>
<td>10.2s</td>
<td>37.2s</td>
</tr>
<tr>
<td>swbd</td>
<td>309</td>
<td>47,578</td>
<td>19.0s</td>
<td>TO</td>
</tr>
<tr>
<td>tiger</td>
<td>318</td>
<td>52,184</td>
<td>94.5s</td>
<td>17.5s</td>
</tr>
<tr>
<td>tuebadz</td>
<td>196</td>
<td>8,932</td>
<td>2.6s</td>
<td>15.3s</td>
</tr>
<tr>
<td>wsj</td>
<td>240</td>
<td>31,170</td>
<td>30.3s</td>
<td>TO</td>
</tr>
</tbody>
</table>

- #decimal digits of numerators/denominators in exact rationals always < 10

QF_NRA aka ETR
### Stochastic Grammars Benchmark

#### Certificates for inconsistency

<table>
<thead>
<tr>
<th>name</th>
<th>non-terminals</th>
<th>rules</th>
<th>time OVI</th>
<th>time SMT (z3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>brown</td>
<td>37</td>
<td>22,866</td>
<td>3.2s</td>
<td>TO</td>
</tr>
<tr>
<td>lemonde</td>
<td>121</td>
<td>32,885</td>
<td>40.1s</td>
<td>TO</td>
</tr>
<tr>
<td>negra</td>
<td>256</td>
<td>29,297</td>
<td>10.2s</td>
<td>37.2s</td>
</tr>
<tr>
<td>swbd</td>
<td>309</td>
<td>47,578</td>
<td>19.0s</td>
<td>TO</td>
</tr>
<tr>
<td>tiger</td>
<td>318</td>
<td>52,184</td>
<td>94.5s</td>
<td>17.5s</td>
</tr>
<tr>
<td>tuebadz</td>
<td>196</td>
<td>8,932</td>
<td>2.6s</td>
<td>15.3s</td>
</tr>
<tr>
<td>wsj</td>
<td>240</td>
<td>31,170</td>
<td>30.3s</td>
<td>TO</td>
</tr>
</tbody>
</table>

- #decimal digits of numerators/denominators in exact rationals always < 10
- ≈ 90% of runtime for arbitrary precision rational arithmetic
Summary & Outlook

- Optimistic “guess-and-check” algorithm for computing self-certifying upper bounds on least solution of positive polynomial equations
- Certified verification of recursive probabilistic system
- Open: Complexity of algorithm
- Follow-up paper: Certificates for lower bounds & termination [W. & Katoen LICS’23]
Summary & Outlook

- Optimistic “guess-and-check” algorithm for computing self-certifying upper bounds on least solution of positive polynomial equations

- Certified verification of recursive probabilistic system

- Open: Complexity of algorithm

- Follow-up paper: Certificates for lower bounds & termination [W. & Katoen LICS’23]

Thank you for listening!
| benchmark     | $|Q|$ | $|P|$ | $|\Gamma|$ | vars | terms | sccs | scc_{max} | cert | $G$ | $D$ | $t_Q$ | $t_{tot}$ | cert_{z3} | $D_{z3}$ | $t_{z3}$ |
|--------------|-----|-----|------|------|------|------|----------|------|-----|-----|------|---------|----------|--------|--------|
| rw-0.499     | 18  | 29  | 5    | 38   | 45   | 1    | 12       | ✓    | 5   | 5   | 17%  | 163     | ✓        | 2      | 11     |
| rw-0.500     | 18  | 29  | 5    | 38   | 45   | 1    | 12       | ×    | 10  | -   | -    | 7327    | ✓        | 2      | 10     |
| rw-0.501     | 18  | 29  | 5    | 38   | 45   | 1    | 12       | ✓    | 5   | 4   | 6%   | 36      | ✓        | 13     | 12     |
| geom-offspring | 24  | 40  | 5    | 52   | 80   | 4    | 24       | ✓    | 8   | 6   | 13%  | 15      | ✓        | 9      | 16     |
| golden       | 27  | 49  | 6    | 81   | 94   | 1    | 36       | ✓    | 1   | 5   | 30%  | 10      | ✓        | 7      | 14     |
| and-or       | 50  | 90  | 7    | 149  | 182  | 1    | 48       | ✓    | 2   | 4   | 26%  | 19      | ✓        | 12     | 15260  |
| gen-fun      | 85  | 219 | 7    | 202  | 327  | 1    | 16       | ✓    | 2   | 3   | 32%  | 22      | ✓        | 15     | 141    |
| virus        | 68  | 149 | 27   | 341  | 551  | 1    | 220      | ✓    | 1   | 5   | 38%  | 40      | ✓        | 7      | 139    |
| escape10     | 109 | 174 | 23   | 220  | 263  | 1    | 122      | ✓    | 1   | 4   | 5%   | 56      | ✓        | 7      | 48     |
| escape25     | 258 | 413 | 53   | 518  | 621  | 1    | 300      | ✓    | 1   | 5   | 17%  | 245     | ✓        | 7      | 15958  |
| escape50     | 508 | 813 | 103  | 1018 | 1221 | 1     | 600      | ✓    | 1   | 7   | 23%  | 653     | ✓        | 7      | 410    |
| escape75     | 760 | 1215| 153  | 1522 | 1825 | 1     | 904      | ✓    | 2   | 9   | 10%  | 3803    | ×        | -      | TO     |
| escape100    | 1009| 1614| 203  | 2020 | 2423 | 1     | 1202     | ×    | 5   | -   | -    | 29027   | ✓        | 6      | 939    |
| escape200    | 2008| 3213| 403  | 4018 | 4821 | 1     | 2400     | ×    | 6   | -   | -    | 83781   | ×        | -      | TO     |
| sequential5  | 230 | 490 | 39   | 1017 | 1200 | 10    | 12       | ✓    | 15  | 4   | 26%  | 103     | ✓        | 8      | 1074   |
| sequential7  | 572 | 1354| 137  | 3349 | 3856 | 14    | 12       | ✓    | 21  | 5   | 27%  | 1049    | ✓        | 8      | 12822  |
| sequential10 | 3341| 8666| 1036 | 26367| 29616| 20    | 12       | ✓    | 30  | 5   | 2%   | 100613  | ✓        | 8      | 453718 |
| mod5         | 44  | 103 | 10   | 296  | 425  | 1     | 86       | ✓    | 1   | 5   | 39%  | 28      | ✓        | 9      | 34150  |
| mod7         | 64  | 159 | 14   | 680  | 1017 | 1     | 222      | ✓    | 1   | 6   | 69%  | 172     | ✓        | 7      | 443    |
| mod10        | 95  | 244 | 20   | 1574 | 2403 | 1     | 557      | ×    | 1   | -   | -    | 675     | ✓        | 7      | 1245   |