Unfolding-Based Partial Order Reduction

César Rodríguez ¹  Marcelo Sousa ²  Subodh Sharma ³  Daniel Kroening ²

¹Université Paris 13, Sorbonne Paris Cité, LIPN, CNRS, France
²Department of Computer Science, University of Oxford, UK
³Indian Institute of Technology Delhi, India

CONCUR’15, Madrid, 4 September 2015
System \models \text{Specification}

\begin{itemize}
  \item Proof
  \item Counterexample
\end{itemize}
Model Checking

System \implies \text{Specification}

State-space

- Proof
- Counterexample

Sources of state-space explosion
- Concurrency
- Nondeterminism
- Data
- Unboundedness
Model Checking

System \models \text{Specification}

- Proof
- Counterexample

Sources of state-space explosion
- Concurrency
- Nondeterminism
- Data
- Unboundedness...
Model Checking

System \( \equiv \) Specification

Sources of state-space explosion

- Concurrency
- Nondeterminism
- Data
- Unboundedness...
Model Checking

System \Rightarrow \equiv \text{Specification}

State-space

- Proof
- Counterexample

Sources of state-space explosion
- Concurrency \rightarrow \text{Partial-order reduction and unfoldings}
- Nondeterminism
- Data
- Unboundedness...
Partial-Order Reductions (PORs) and Unfoldings

**POR**: large family of techniques, interleaving semantics
- Scope of this work: explicit-state, independence-based PORs for reachability
- Persistent sets [Godefroid 96], and early stubborn sets [Valmari 91]

**Unfoldings**: partial-order semantics
- Mainly for Petri nets
- Processes [Petri 66], event structures [Winskel 87], finite prefixes [McMillan 92]

Quite independent fields of research for the last 20 years

To what extent both approaches
1. exploit the same source of reduction?
2. are just different exploration algorithms over the same underlying reduced space?
3. How can that be exploited to improve both methods?
Partial-Order Reductions (conceptually)

- Independence relation $\diamond \subseteq T \times T$
- At least one $\diamond$-equivalent run to every terminating run
- POR is optimal if exactly one
The Unfolding Approach (conceptually)

\[ M \overset{\parallel}{=} \text{Unfolding} \]

\[ \text{runs} (M) \overset{\parallel}{=} \text{po-runs} (M) \overset{\parallel}{=} U_M \]

\( \langle T, \ldots \rangle \) (Petri net)
The Unfolding Approach (conceptually)

A labelled prime event structure is a tuple \( \langle E, <, \#, \lambda \rangle \) where

- \( E \) is the set of events, labelled by \( \lambda : E \to T \)
- \( e < e' \iff e' \) occurs \( \Rightarrow e \) occurs before (causality)
- \( e \# e' \iff e \) and \( e' \) cannot occur in same execution (conflict)

Intuition: a set of partial orders bound together by \( \# \) in a prefix-sharing fashion:

A configuration is any set \( C \subseteq E \) s.t:

- if \( e \in C \) and \( e' < e \), then \( e' \in C \) (causally closed)
- no two events in \( C \) are in conflict (conflict free)

Intuition: configuration \( \equiv \) constituent partial order (or prefix of)
The Unfolding Approach (conceptually)

Unfolding

\[ M \xrightarrow{\parallel} \text{runs }(M) \xrightarrow{} \text{po-runs }(M) \xrightarrow{} \mathcal{U}_M \]

\((\langle T, \ldots \rangle)\) (Petri net)
The Unfolding Approach (conceptually)

First contribution:
- Unfolding semantics for general model of computation $M := \langle T, \ldots \rangle$
- Parametric on user-defined independence relation $\Diamond \subseteq T \times T$
Let

- $M := \langle T, \ldots \rangle$ be a general model of computation
- $\diamond \subseteq T \times T$ a valid unconditional independence relation

Theorem (Soundness & Uniqueness)

- The unfolding $\mathcal{U}_M,\diamond$ exists and is unique.
- Every sequential execution is represented by exactly 1 configuration.
Unfolding over User-Defined Independence Relation

Let

- $M := \langle T, \ldots \rangle$ be a general model of computation
- $\bowtie \subseteq T \times T$ a valid unconditional independence relation

<table>
<thead>
<tr>
<th>$w$</th>
<th>$r$</th>
<th>$r'$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x=1$</td>
<td>$y=x$</td>
<td>$z=x$</td>
</tr>
</tbody>
</table>

$T = \{w, r, r'\}$

$\bowtie = \{(r, r')\}$

Partially-ordered runs:

**Theorem (Soundness & Uniqueness)**

- The unfolding $U_{M, \bowtie}$ exists and is unique.
- Every sequential execution is represented by exactly 1 configuration.
Unfolding over User-Defined Independence Relation

Let

- $M := \langle T, \ldots \rangle$ be a general model of computation
- $\diamond \subseteq T \times T$ a valid unconditional independence relation

<table>
<thead>
<tr>
<th>$w$</th>
<th>$r$</th>
<th>$r'$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x = 1$</td>
<td>$y = x$</td>
<td>$z = x$</td>
</tr>
</tbody>
</table>

$T = \{w, r, r'\}$

$\diamond = \{ (r, r') \}$

Theorem (Soundness & Uniqueness)

- The unfolding $U_M, \diamond$ exists and is unique.
- Every sequential execution is represented by exactly 1 configuration.
Motivating Questions

Do both approaches exploit the same source of reduction?

- Yes, independence-based PORs and $U_M,\diamond$ solely exploit $\diamond$
- $\mathcal{R}_M,\diamond$ is polynomial-time convertible into $U_M,\diamond$

Are they just different exploration algorithms over the same underlying reduced space?

- Necessarily yes!
- In the sequel: 6 algorithmic differences

Main contribution

A novel stateless POR exploration of unfolding semantics

- Retains advantages of both approaches
- (Super-)Optimal: can explore fewer executions than Mazurkiewicz traces
POR vs Unfoldings: 6 Algorithmic Differences

1. Unfolding extension is NP-complete; POR extension is constant-time
POR vs Unfoldings: 6 Algorithmic Differences

1. Unfolding extension is **NP-complete**; POR extension is **constant-time**

2. $R_{M,\diamond}$ can be **exponentially larger** than $U_{M,\diamond}$
POR vs Unfoldings: 6 Algorithmic Differences

1. Unfolding extension is \textbf{NP-complete}; POR extension is \textbf{constant-time}

2. $\mathcal{R}_M,\diamond$ can be \textbf{exponentially larger} than $U_M,\diamond$

3. Unfolding algorithms are inherently \textbf{stateful}; state-of-the-art DPORs are \textbf{stateless}

   - [Flanagan, Godefroid, POPL’05], [Abdulla et al., POPL’14]

4. Dynamic POR: difficult to avoid repeated exploration of same states

5. Dynamic POR: difficult to handle non-terminating executions

6. Stateless PORs do not profit from additional RAM
1. Unfolding extension is **NP-complete**; POR extension is **constant-time**

2. $R_{M,\Diamond}$ can be **exponentially larger** than $U_{M,\Diamond}$

3. Unfolding algorithms are inherently **stateful**; state-of-the-art DPORs are **stateless**

4. Dynamic POR: difficult to avoid repeated exploration of **same states**
POR vs Unfoldings: 6 Algorithmic Differences

1. Unfolding extension is **NP-complete**; POR extension is **constant-time**

2. $\mathcal{R}_{M,\diamond}$ can be exponentially larger than $\mathcal{U}_{M,\diamond}$

3. Unfolding algorithms are inherently **stateful**; state-of-the-art DPORs are **stateless**

4. Dynamic POR: difficult to avoid repeated exploration of **same states**

5. Dynamic POR: difficult to handle **non-terminating executions**
POR vs Unfoldings: 6 Algorithmic Differences

1. Unfolding extension is **NP-complete**; POR extension is **constant-time**

2. $R_M,\diamond$ can be **exponentially larger** than $U_M,\diamond$

3. Unfolding algorithms are inherently **stateful**; state-of-the-art DPORs are **stateless**

4. Dynamic POR: difficult to avoid repeated exploration of **same states**

5. Dynamic POR: difficult to handle **non-terminating executions**

6. Stateless PORs do not profit from **additional RAM**
POR vs Unfoldings: 6 Algorithmic Differences

1. Unfolding extension is NP-complete; POR extension is constant-time
2. $R_M, \Diamond$ can be exponentially larger than $U_M, \Diamond$
3. Unfolding algorithms are inherently stateful; state-of-the-art DPORs are stateless
4. Dynamic POR: difficult to avoid repeated exploration of same states
5. Dynamic POR: difficult to handle non-terminating executions
6. Stateless PORs do not profit from additional RAM
1. Unfolding extension is NP-complete; POR extension is constant-time

2. $R_M,\diamond$ can be exponentially larger than $U_M,\diamond$

3. Unfolding algorithms are inherently stateful; state-of-the-art DPORs are stateless

4. Dynamic POR: difficult to avoid repeated exploration of same states

5. Dynamic POR: difficult to handle non-terminating executions

6. Stateless PORs do not profit from additional RAM

Our algorithm (main contribution, next slide)

A novel stateless POR exploration of unfolding semantics

- Retains advantages of both approaches
- (Super-)Optimal: can explore fewer executions than Mazurkiewicz traces
- Addresses all above points except (2)
Stateless Unfolding-based Partial-Order Reduction

\[ \begin{array}{ccc}
   w & r & r' \\
   x=1 & y=x & z=x
\end{array} \]

\[ C = \{ \} \]
\[ e = 1 \]
\[ A = \{ \} \]
\[ U = \{ \} \]
\[ G = \{ \} \]
Stateless Unfolding-based Partial-Order Reduction

\[ w \quad r \quad r' \]
\[ x=1 \quad y=x \quad z=x \]

\[
\begin{array}{c}
1 \\
2 \\
3 \\
4 \\
5 \\
6 \\
7 \\
8 \\
9 \\
10 \\
\end{array}
\]

\[
\begin{array}{c}
\{1\} \\
\{1, 2\} \\
\{1, 2, 3\} \\
\{1, 2, 3, 4\} \\
\{1, 2, 3, 4, 5\} \\
\{1, 2, 3, 4, 5, 6\} \\
\{1, 2, 3, 4, 5, 6, 7\} \\
\{1, 2, 3, 4, 5, 6, 7, 8\} \\
\{1, 2, 3, 4, 5, 6, 7, 8, 9\} \\
\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\} \\
\end{array}
\]

\[
\begin{array}{c}
C = \{1\} \\
U = \{1\} \\
G = \{} \\
e = 2 \\
A = \{}
\end{array}
\]
Stateless Unfolding-based Partial-Order Reduction

\[
\begin{align*}
\begin{array}{ccc}
 w & r & r' \\
 x=1 & y=x & z=x \\
\end{array}
\end{align*}
\]

C = \{1, 2\} \quad e = 3 \quad A = \{
\}

U = \{1, 2\} \quad G = \{
\}
Stateless Unfolding-based Partial-Order Reduction

\[ C = \{1, 2, 3\} \quad e = \text{n/a} \quad A = \{\}\] 

\[ U = \{1, 2, 3, 4, 7\} \quad G = \{\}\]
Stateless Unfolding-based Partial-Order Reduction

\[ \begin{array}{ccc}
  w & r & r' \\
  x=1 & y=x & z=x \\
\end{array} \]

\begin{itemize}
  \item \( C = \{1, 2\} \)
  \item \( e = 3 \)
  \item \( A = \{\} \)
  \item \( U = \{1, 2, 4, 7\} \)
  \item \( G = \{3\} \)
\end{itemize}
Stateless Unfolding-based Partial-Order Reduction

\[
\begin{array}{ccc}
 w & r & r' \\
x = 1 & y = x & z = x \\
\end{array}
\]

\[
\begin{array}{c}
1 \quad 4 \\
3 \quad 8 \\
5 \quad 9 \\
2 \quad 6 \\
7 \quad 10 \\
\end{array}
\]

\[
\begin{array}{c}
\{1\} \\
\{1, 2\} \\
\{1, 2, 3\} \\
\{4\} \\
\{4, 5\} \\
\{4, 5, 6\} \\
\{4, 7\} \\
\{4, 7, 8\} \\
\{7, 9\} \\
\{7, 9, 10\} \\
\end{array}
\]

\[
\begin{array}{c}
C = \{1\} \\
U = \{1, 4, 7\} \\
G = \{3, 2\} \\
e = 2 \\
A = \{\} \\
\end{array}
\]
Stateless Unfolding-based Partial-Order Reduction

\[
\begin{array}{ccc}
\omega & r & r' \\
x &= 1 & y = x & z = x
\end{array}
\]

\[
\begin{array}{cccc}
\omega & 1 & 4 & 7 \\
2 & 3 & 5 & 8 & 9 & w \\
r & r' & w & w & w \\
6 & 10 & r & r' & r
\end{array}
\]

\[
\begin{array}{cccc}
1 & 2 & 4 & \{1\} \\
1, 2, 3 & 4, 5 & 4 & 7, 9 \{7\} \\
4, 5, 6 & 4, 7 & 7, 9 & 7, 9, 10
\end{array}
\]

\[
C = \{\} \quad e = 1 \quad A = \{\} \\
U = \{4, 7\} \quad G = \{3, 2\}
\]
Stateless Unfolding-based Partial-Order Reduction

\[ \begin{align*}
  w & \quad r & \quad r' \\
  x = 1 & \quad y = x & \quad z = x \\
\end{align*} \]

\begin{align*}
  w & \quad r & \quad r' \\
  1 & \quad 4 & \quad 7 \\
  2 & \quad 3 & \quad 5 \\
  6 & \quad 8 & \quad 9 \\
  10 & \\
\end{align*}

C = \{\} \quad e = 4 \quad A = \{4\}

U = \{4, 7, 1\} \quad G = \{3, 2\}
Stateless Unfolding-based Partial-Order Reduction

\[
\begin{array}{ccc}
  w & r & r' \\
  x=1 & y=x & z=x \\
\end{array}
\]

\[
\begin{array}{c}
  \{1\} \\
  \{1, 2\} \\
  \{1, 2, 3\} \\
  \{4\} \\
  \{4, 5\} \\
  \{4, 5, 6\} \\
  \{4, 7\} \\
  \{4, 7, 8\} \\
  \{7\} \\
  \{7, 9\} \\
  \{7, 9, 10\} \\
\end{array}
\]

\[
C = \{4\} \quad e = 5 \quad A = \{
\}
\]

\[
U = \{4, 7, 1\} \quad G = \{3, 2\}
\]
Stateless Unfolding-based Partial-Order Reduction

\[ w \quad r \quad r' \]
\[ x = 1 \quad y = x \quad z = x \]

\[ C = \{4, 5\} \quad e = 6 \quad A = \{\} \]
\[ U = \{4, 5, 7, 1\} \quad G = \{3, 2\} \]
Stateless Unfolding-based Partial-Order Reduction

\[
\begin{array}{ccc}
  w & r & r' \\
  x=1 & y=x & z=x \\
\end{array}
\]

\[
\begin{array}{ccc}
  \downarrow & \downarrow & \downarrow \\
  1 & 4 & 7 \\
  2 & 3 & 5 \\
  8 & 9 & 10 \\
\end{array}
\]

\[
\begin{array}{ccc}
  \downarrow & \downarrow & \downarrow \\
  \{4,5,6\} & \{4,7\} & \{7,9,10\} \\
  \{1,2,3\} & \{4,5\} & \{4\} \\
  \{1\} & \{4\} & \{7\} \\
  \{} & \{} & \{} \\
\end{array}
\]

\[
\begin{align*}
C &= \{4,5,6\} \\
U &= \{4,5,6,7,1\} \\
G &= \{3,2\} \\
e &= n/a \\
A &= \{} \\
\end{align*}
\]
Stateless Unfolding-based Partial-Order Reduction

\[
\begin{array}{ccc}
  w & r & r' \\
  x=1 & y=x & z=x \\
\end{array}
\]

C. Rodríguez, M. Sousa, S. Sharma, D. Kroening

C = \{4, 5\} \quad e = 6 \quad A = \{
U = \{4, 5, 7, 1\} \quad G = \{3, 2, 6\}
Stateless Unfolding-based Partial-Order Reduction

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(w)</td>
<td>(r)</td>
<td>(r')</td>
</tr>
<tr>
<td>x=1</td>
<td>y=x</td>
<td>z=x</td>
</tr>
</tbody>
</table>

\[
\begin{array}{c}
\text{w} \\
\text{r} \\
\text{r'} \\
\end{array}
\]

\[
\begin{array}{c}
\text{1} \\
\text{2} \\
\text{3} \\
\text{4} \\
\text{5} \\
\text{6} \\
\text{7} \\
\text{8} \\
\text{9} \\
\text{10} \\
\end{array}
\]

\[
\begin{array}{c}
\{\} \\
\{1\} \\
\{1, 2\} \\
\{1, 2, 3\} \\
\{1, 2, 3, 4\} \\
\{1, 2, 3, 4, 5\} \\
\{1, 2, 3, 4, 5, 6\} \\
\{1, 2, 3, 4, 5, 6, 7\} \\
\{1, 2, 3, 4, 5, 6, 7, 9\} \\
\{1, 2, 3, 4, 5, 6, 7, 9, 10\} \\
\end{array}
\]

\[
\begin{array}{c}
\{4\} \\
\{4\} \\
\{4\} \\
\{4\} \\
\{4\} \\
\{4\} \\
\{4\} \\
\{4\} \\
\{7\} \\
\{7\} \\
\end{array}
\]

\[
\begin{array}{c}
e = 5 \\
A = \{\} \\
C = \{4\} \\
U = \{4, 7, 1\} \\
G = \{3, 2, 6\}
\end{array}
\]
Stateless Unfolding-based Partial-Order Reduction

The diagram illustrates the unfolding-based partial-order reduction with the following transitions:

- **w**: 1 → 4 → 7 → 6 → 10
- **r**: x = 1
- **r’**: y = x, z = x

The figure shows the states and transitions with the following sets:

- \( C = \{4\} \)
- \( e = 7 \)
- \( A = \{7\} \)
- \( U = \{4, 7, 1, 5\} \)
- \( G = \{3, 2, 6\} \)
\[ w \quad r \quad r' \]
\[ x = 1 \quad y = x \quad z = x \]

\[ \begin{align*}
C &= \{4, 7\} \\
U &= \{4, 7, 1, 5\} \\
G &= \{3, 2, 6\} \\
e &= 8 \\
A &= \{\} 
\end{align*} \]
Stateless Unfolding-based Partial-Order Reduction

\[\begin{array}{ccc}
w & r & r' \\
x=1 & y=x & z=x \\
\end{array}\]

\[
\begin{array}{c}
\begin{tikzpicture}
\node (1) at (0,0) {$1$};
\node (2) at (-1,-1) {$2$};
\node (3) at (1,-1) {$3$};
\node (4) at (-2,-2) {$w$};
\node (5) at (0,-2) {$5$};
\node (6) at (1,-2) {$w$};
\node (7) at (2,-2) {$7$};
\node (8) at (-2,-3) {$r$};
\node (9) at (0,-3) {$8$};
\node (10) at (1,-3) {$9$};
\node (11) at (2,-3) {$w$};
\node (12) at (3,-3) {$r$};
\node (13) at (0,-4) {$w$};
\node (14) at (1,-4) {$10$};
\node (15) at (2,-4) {$r$};
\node (16) at (3,-4) {$\bot$};
\draw (1) -- (2);
\draw (1) -- (3);
\draw (1) -- (4);
\draw (2) -- (4);
\draw (3) -- (4);
\draw (4) -- (5);
\draw (4) -- (6);
\draw (5) -- (8);
\draw (6) -- (8);
\draw (6) -- (9);
\draw (7) -- (10);
\draw (10) -- (15);
\end{tikzpicture}
\end{array}
\]

\[
\begin{array}{c}
\begin{tikzpicture}
\node (1) at (0,0) {$\{\}$};
\node (2) at (-1,-1) {$\{1\}$};
\node (3) at (0,-2) {$\{1,2\}$};
\node (4) at (1,-1) {$\{4\}$};
\node (5) at (1,-2) {$\{4,5\}$};
\node (6) at (-2,-2) {$\{1,2,3\}$};
\node (7) at (0,-3) {$\{4,5,6\}$};
\node (8) at (1,-3) {$\{4,7\}$};
\node (9) at (2,-3) {$\{7,9\}$};
\node (10) at (3,-3) {$\{7,9,10\}$};
\node (11) at (4,-3) {$\{\}$};
\node (12) at (0,-4) {$\{1\}$};
\node (13) at (1,-4) {$\{4\}$};
\node (14) at (2,-4) {$\{4,7,8\}$};
\node (15) at (3,-4) {$\{7,9\}$};
\node (16) at (4,-4) {$\{7,9,10\}$};
\node (17) at (5,-4) {$\{\}$};
\node (18) at (6,-4) {$\{\}$};
\node (19) at (0,-5) {$\{1\}$};
\node (20) at (1,-5) {$\{4\}$};
\node (21) at (2,-5) {$\{4,7,8\}$};
\node (22) at (3,-5) {$\{7,9\}$};
\node (23) at (4,-5) {$\{7,9,10\}$};
\node (24) at (5,-5) {$\{\}$};
\node (25) at (6,-5) {$\{\}$};
\node (26) at (0,-6) {$\{1\}$};
\node (27) at (1,-6) {$\{4\}$};
\node (28) at (2,-6) {$\{4,7,8\}$};
\node (29) at (3,-6) {$\{7,9\}$};
\node (30) at (4,-6) {$\{7,9,10\}$};
\node (31) at (5,-6) {$\{\}$};
\node (32) at (6,-6) {$\{\}$};
\node (33) at (0,-7) {$\{1\}$};
\node (34) at (1,-7) {$\{4\}$};
\node (35) at (2,-7) {$\{4,7,8\}$};
\node (36) at (3,-7) {$\{7,9\}$};
\node (37) at (4,-7) {$\{7,9,10\}$};
\node (38) at (5,-7) {$\{\}$};
\node (39) at (6,-7) {$\{\}$};
\node (40) at (0,-8) {$\{1\}$};
\node (41) at (1,-8) {$\{4\}$};
\node (42) at (2,-8) {$\{4,7,8\}$};
\node (43) at (3,-8) {$\{7,9\}$};
\node (44) at (4,-8) {$\{7,9,10\}$};
\node (45) at (5,-8) {$\{\}$};
\node (46) at (6,-8) {$\{\}$};
\end{tikzpicture}
\end{array}
\]

\[
C = \{4,7,8\} \quad e = n/a \quad A = \{}
\]

\[
U = \{4,7,8,1,5\} \quad G = \{3,2,6\}
\]
Stateless Unfolding-based Partial-Order Reduction

\[ \begin{align*}
   w & \quad r & \quad r' \\
   x=1 & \quad y=x & \quad z=x \\
\end{align*} \]

\[
\begin{array}{lll}
   w & r & r' \\
   \quad 1 & \quad 4 & \quad 7 \\
   \quad 2 & \quad 3 & \quad 5 \\
   \quad 8 & \quad 9 & \quad 10 \\
\end{array}
\]

\[
\begin{array}{lll}
   w & r & r' \\
   \quad 4 & \quad 5 & \quad 6 \\
   \quad 8 & \quad 9 & \quad 10 \\
\end{array}
\]

\[
\begin{array}{lll}
   w & r & r' \\
   \quad 4 & \quad 7 & \quad 8 \\
   \quad 9 & \quad 10 & \quad \text{C. Rodríguez, M. Sousa, S. Sharma, D. Kroening}
\end{array}
\]

\[
\begin{array}{lll}
   w & r & r' \\
   \quad 7 & \quad 6 & \quad \text{Unfolding-Based Partial Order Reduction}
\end{array}
\]

\[
C = \{4, 7\} \quad e = 8 \quad A = \{\}
\]

\[
U = \{4, 7, 1, 5\} \quad G = \{3, 2, 6, 8\}
\]
Stateless Unfolding-based Partial-Order Reduction

\[ \begin{array}{ccc}
\text{w} & \text{r} & \text{r'} \\
\text{x=1} & \text{y=x} & \text{z=x} \\
\end{array} \]

\[
\begin{array}{c}
\downarrow \\
1 \quad 4 \\
\{1\} \quad \{4\} \\
\{1, 2\} \quad \{4, 5\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\{1, 2, 3\} \quad \{4, 5, 6\} \\
\} = \{4\} \\
\} = 7 \\
\} = \{\} \\
\]
Stateless Unfolding-based Partial-Order Reduction

\[
\begin{array}{ccc}
  w & r & r' \\
  x=1 & y=x & z=x \\
\end{array}
\]

\[
\begin{array}{c}
\downarrow \\
1 \\
\downarrow \\
\{1\} \\
\downarrow \\
2 \\
\downarrow \\
\{1, 2\} \\
\downarrow \\
3 \\
\downarrow \\
\{1, 2, 3\} \\
\downarrow \\
4 \\
\downarrow \\
\{4\} \\
\downarrow \\
5 \\
\downarrow \\
\{4\} \\
\downarrow \\
6 \\
\downarrow \\
\{4, 5, 6\} \\
\downarrow \\
7 \\
\downarrow \\
\{7\} \\
\end{array}
\]

\[
C = \{4\} \quad e = 5 \quad A = \{\}
\]

\[
U = \{4, 7, 1\} \quad G = \{3, 2, 6, 8\}
\]
Stateless Unfolding-based Partial-Order Reduction

<table>
<thead>
<tr>
<th>w</th>
<th>r</th>
<th>r'</th>
</tr>
</thead>
<tbody>
<tr>
<td>x=1</td>
<td>y=x</td>
<td>z=x</td>
</tr>
</tbody>
</table>

C = {}  
U = {4, 7, 1}  
G = {3, 2, 6, 8}  
e = 4  
A = {}
Stateless Unfolding-based Partial-Order Reduction

\[ w \quad r \quad r' \]
\[ x = 1 \quad y = x \quad z = x \]

\[
\begin{array}{c}
1 \quad 4 \quad 7 \\
2 \quad 5 \quad 8 \\
3 \quad 6 \quad 9 \\
\end{array}
\]

\[
\begin{array}{c}
\{1\} \quad \{4\} \\
\{1, 2\} \quad \{4, 5\} \\
\{1, 2, 3\} \quad \{4, 7\} \\
\{4, 5, 6\} \quad \{7, 9\} \\
\{4, 7, 8\} \quad \{7, 9, 10\} \\
\end{array}
\]

\[ C = \{\} \]
\[ e = 7 \]
\[ A = \{7, 9\} \]
\[ U = \{7, 1, 4\} \]
\[ G = \{3, 2, 6, 8\} \]
Stateless Unfolding-based Partial-Order Reduction

\[ \begin{array}{ccc}
  w & r & r' \\
  x=1 & y=x & z=x \\
\end{array} \]

\[
\begin{array}{llllllllll}
& w & & r & & r' & & w & & r & & r' & \\
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & \\
\end{array}
\]

\[
\begin{array}{llllllllll}
& & w & & r & & r' & & w & & r & & r' & \\
\end{array}
\]

\[
\begin{array}{llllllllll}
& & & & & w & & \{4\} & & \{7\} & \\
\end{array}
\]

\[
\begin{array}{llllllllll}
& & & & & & & & & e = 9 & \\
\end{array}
\]

\[
\begin{array}{llllllllll}
& & & & & & & & & A = \{9\} & \\
\end{array}
\]

\[
\begin{array}{llllllllll}
& & & & & & & & & G = \{3, 2, 6, 8\} & \\
\end{array}
\]

\[
\begin{array}{llllllllll}
& & & & & & & & & C = \{7\} & \\
\end{array}
\]

\[
\begin{array}{llllllllll}
& & & & & & & & & U = \{7, 1, 4\} & \\
\end{array}
\]
Stateless Unfolding-based Partial-Order Reduction

\[
\begin{array}{ccc}
  w & r & r' \\
  x &=& 1 \\
  y &=& x \\
  z &=& x \\
\end{array}
\]

\[
\begin{array}{c}
  1 \\
  4 \\
  7 \\
\end{array}
\]

\[
\begin{array}{c}
  2 \\
  3 \\
  5 \\
  8 \\
  9 \\
\end{array}
\]

\[
\begin{array}{c}
  6 \\
  10 \\
\end{array}
\]

\[
\begin{array}{c}
  1 \\
  2 \\
  3 \\
  4 \\
  5 \\
  6 \\
  7 \\
  8 \\
  9 \\
  10 \\
\end{array}
\]

\[
C = \{7, 9\} \quad e = 10 \quad A = \{\}
\]

\[
U = \{7, 9, 1, 4\} \quad G = \{3, 2, 6, 8\}
\]
Stateless Unfolding-based Partial-Order Reduction

\[
\begin{array}{ccc}
  w & r & r' \\
  x=1 & y=x & z=x \\
\end{array}
\]

\[
\begin{array}{cccc}
  & w & r & r' \\
  & 1 & 4 & 7 \\
  & 2 & 3 & 5 & 8 & 9 & 6 & 10 \\
\end{array}
\]

\[
\begin{array}{cccc}
  & {} & 1 & \} \\
  & 1 & 2 & \} \\
  & 4 & 5 & \} \\
  \} & 3 & 4 & \} \\
  \} & 6 & 7 & \} \\
  \} & 8 & 9 & \} \\
  \} & 10 & \} \\
\end{array}
\]

\[
C = \{7, 9, 10\} \quad e = n/a \quad A = \{\} \\
U = \{7, 9, 10, 1, 4\} \quad G = \{3, 2, 6, 8\}
\]
For terminating systems (acyclic state-space) we have:

Theorem

The previous algorithm:

1. Always stops (termination)
2. Explores at least once every maximal configuration of $U_M,\Diamond$ (completeness)
3. Never explores twice any intermediate/maximal configuration (optimality)

Next: we extend the algorithm to handle non-terminating executions
while (1):
    lock(m)
    if (buf < MAX): buf++
    unlock(m)
while (1):
    lock(m)
    if (buf > MIN): buf--
    unlock(m)

Thread 1
l, b+, u, l, b+, u, l, b--

Thread 2
(m = 0, b = 1)

Thread 1
l, b+
Cutoffs – Intuitions

while (1):
  lock(m)
  if (buf < MAX): buf++
  unlock(m)

while (1):
  lock(m)
  if (buf > MIN): buf--
  unlock(m)

Thread 1
- l, b+, u, l, b+, u, l, b-

Thread 2
- (m = 0, b = 1)
  lock
  buf++
  unlock
  lock
  buf++
  unlock
  lock
  buf--
Cutoffs – Intuitions

while (1):
    lock(m)
    if (buf < MAX): buf++
    unlock(m)

while (1):
    lock(m)
    if (buf > MIN): buf--
    unlock(m)

Thread 1
- l, b+, u, l, b+, u, l, b-

Thread 2
- m = 0, b = 1

Event $e$ of $U_M,\Diamond$ is cutoff if $U_M,\Diamond$ contains a corresponding event $e'$ s.t.:
- $\text{state}([e]) = \text{state}([e'])$
- $|[e']| < |[e]|$
Cutoffs – Intuitions

while (1):
  lock(m)
  if (buf < MAX): buf++
  unlock(m)

while (1):
  lock(m)
  if (buf > MIN): buf--
  unlock(m)

Thread 1
l, b+, u, l, b+, u, l, b–

Thread 2
(m = 0, b = 1)

Thread 1
l, b+

Local configuration: \([e] := \{e' \in E : e' < e\}\)

Event \(e\) of \(U_{M,\Diamond}\) is cutoff if \(U_{M,\Diamond}\) contains a corresponding event \(e'\) s.t.:

\begin{itemize}
  \item \(state([e]) = state([e'])\)
  \item \(|[e']| < |[e]|\)
\end{itemize}
Cutoffs – Application to our Algorithm

Problem: Preventing the algorithm from getting stuck in infinite executions
Solution: To explore event $e$ if predicate $cutoff(e, U, G)$ evaluates to false

Definition

Predicate $cutoff(e, U, G)$ holds iff there is some $e' \in U \cup G$ s.t.

- $\text{state}([e]) = \text{state}([e'])$
- $|[e']| < |[e]|$

- $cutoff(e, U, G)$ evaluates differently on different executions (!!!)

Theorem (Completeness)

For each fireable transition $t$ of a (possibly) non-terminating system $M$, the modified algorithm explores at least one $t$-labelled event.
State Caching

- Garbage-collected events in $G$ can be cleaned at discretion $\Rightarrow$ cache memory

Cached events exploited in two ways:

1. **Avoid reconstruction** (causality, conflict) when reinserting in $U$ (up to $O(2^n)$ times!)

2. Improve chances of cutting off earlier the unfolding

By contrast:

- Most efficient PORs up to date are stateless (underuse memory)
- Unfolding approaches are stateful (overload memory)
New tool: **POET** (Partial-Order Exploration Tool)

- Deterministic C programs, POSIX threads
- Prototype in Haskell

Goals of the experiments:

- To investigate **characteristics** of average program unfoldings (depth, width)
- Frequency and **impact of cutoffs**
- Compare with **NIDHUGG**, quasi-optimal recent DPOR [Abdulla et al., POPL’14]

Benchmarks:

- Mainly come from the SV-COMP’14

Tool and experiments available online:

http://www.cs.ox.ac.uk/people/marcelo.sousa/poet/
### Experiments — Acyclic State-Space

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>NIDHUGG</th>
<th>POET</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>STF</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>STF</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>SPIN08</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>FIB</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>FIB</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>CCNF(9)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>CCNF(19)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>SSB(1)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>SSB(4)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>SSB(8)</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Remarks:
- Narrow, deep, relatively small unfoldings
- Half of the benchmarks display no concurrency (STF, SPIN08, Fib)
- In SSB we achieve a **super-optimal exploration**
## Experiments — Non-acyclic State-Space

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>NIDHUGG</th>
<th>PoET</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td>$</td>
<td>P</td>
</tr>
<tr>
<td>Szymanski</td>
<td>3</td>
<td>--</td>
</tr>
<tr>
<td>Dekker</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>Lamport</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>Peterson</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>PGSQL</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>RWlock</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>RWlock(2)*</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>Prodc Cons</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Prodc Cons(2)</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

**Remarks:**

- **PoET:** complete verification; **NIDHUGG:** bounded verification
- Significant, sometimes dramatic, reduction in nr. of executions
Concluding Remarks

Results:
- **Parametrized unfolding semantics** exploiting provided independence relation
- **Super-optimal POR** algorithm, handling non-terminating executions
- **State caching**: compromise between stateless and stateful

Consequences:
- Identifies **additional structure** in the POR tree
- Indirectly maps key POR notions to well-established unfolding theory
- Makes possible comparison between unfoldings and PORs for, e.g., programs

Future work:
- Exploiting additional structure