On-the-fly Confluence Detection for Statistical Model Checking

\[ \nu(S_i) = \nu(s_i) \land (S_i = \{ s_i \} \lor \forall s \in S) \]

Arnd Hartmanns and Mark Timmer
Saarland University, Germany
University of Twente, The Netherlands
Introduction

Model Checking

Model Checking

Model

Requirements

model-based testing

System/implementation

Problem: State-space explosion memory consumption

Probabilistic Model Checking

DTMC models: $1 \rightarrow \text{reset} \quad P(\diamond \text{msg}_{rcvd}) \geq 0.99$

Problem: State-space explosion plus numerical complexity
Introduction

Statistical Model Checking (SMC)

SMC = Simulation + Statistics

...confidence intervals, Chernoff-Hoeffding bound, SPRT...
error bounds: e.g. result is $\epsilon$-correct with probability $\delta$

+ constant memory usage (store only current state)
no numeric surprises (e.g. with imprecise arithmetics)
- runtime strongly dependent on desired accuracy
On-the-fly Confluence Detection for SMC

Arnd Hartmanns & Mark Timmer

Introduction

Statistical Model Checking versus Nondeterminism

MDP models:

need to resolve nondeterministic choices

Standard technique:

 implicit uniformly distributed resolution

some value $\in [P_{\text{min}}, P_{\text{max}}]$

widely implemented: PRISM, UPPAAL, ...

$P_{\text{min}} (\diamond \text{msg}_{\text{rcvd}}) \geq 0.99$

$P_{\text{max}} (\diamond \text{msg}_{\text{rcvd}}) \geq 1$
Introduction

Previous approaches to SMC for MDPs

"POR"

Partial order reduction-based method:

Nondeterminism may be **spurious**

= irrelevant for the results, i.e. $P_{\text{max}} = P_{\text{min}}$

⇒ check for spuriousness on-the-fly and ignore

+ very low memory overhead

no change to SMC error bounds

- spurious interleavings only
Introduction

Previous approaches to SMC for MDPs

Learning-based method:

- Use reinforcement learning to obtain memoryless scheduler using simulation
- Use that scheduler for SMC for $P/_{\text{max}}$ (bounded LTL)

- Works for every MDP
- Memory usage to store scheduler
- No error bounds, converges to actual result only

Henriques, Martins, Zuliani, Platzer, Clarke: Statistical Model Checking for Markov Decision Processes (QEST 2012)
Outline

In this talk: a new method based on on-the-fly confluence detection

1 Probabilistic Confluence
   Adaption to SMC & advantages over POR (MT)

2 On-the-fly Detection
   A correct algorithm for use during simulation (MT)

3 Evaluation
   Tools, applicability, performance

Invisible transitions connecting equivalent states

**Invisible** transitions in confluence reduction:

- Deterministic
- Stuttering
Invisible transitions connecting equivalent states

**Invisible** transitions in confluence reduction:

- Deterministic
- Stuttering

Invisible transitions *might* disable behaviour...
Invisible transitions connecting equivalent states

**Invisible** transitions in confluence reduction:
- Deterministic
- Stuttering

Invisible transitions **might** disable behaviour...
Invisible transitions connecting equivalent states

Invisible transitions in confluence reduction:
- Deterministic
- Stuttering

Invisible transitions might disable behaviour...
...though often, they connect equivalent states
Invisible transitions connecting equivalent states

Invisible transitions in confluence reduction:

- Deterministic
- Stuttering

Invisible transitions might disable behaviour...

...though often, they connect equivalent states
Invisible transitions connecting equivalent states

Invisible transitions in confluence reduction:

- Deterministic
- Stuttering

Invisible transitions might disable behaviour...
...though often, they connect equivalent states
Invisible transitions connecting equivalent states

Invisible transitions in confluence reduction:

- Deterministic
- Stuttering

Invisible transitions might disable behaviour...

...though often, they connect equivalent states

\[
\begin{array}{c}
\text{b} \\
\downarrow \\
\text{a} \\
\downarrow \\
\text{c} \\
\end{array}
\quad
\begin{array}{c}
\text{c} \\
\downarrow \\
\text{a} \\
\downarrow \\
\text{b} \\
\end{array}
\quad
\begin{array}{c}
\text{c} \\
\downarrow \\
\text{a} \\
\downarrow \\
\text{b} \\
\end{array}
\quad
\begin{array}{c}
\text{c} \\
\downarrow \\
\text{a} \\
\downarrow \\
\text{b} \\
\end{array}
\]
Non-probabilistic and probabilistic confluence reduction

Confluence reduction:

denoting a subset of the invisible transitions as confluent.
Non-probabilistic and probabilistic confluence reduction

Confluence reduction:
denoting a subset of the invisible transitions as confluent.

Non-probabilistically:
Confluence reduction:
denoting a \textit{subset of the invisible transitions} as confluent.

Non-probabilistically:

\begin{align*}
\begin{array}{c}
\bullet \\
\scriptstyle a \\
\bullet
\end{array}
\end{align*}
\begin{align*}
\begin{array}{c}
\bullet \\
\scriptstyle b \\
\bullet
\end{array}
\end{align*}
\begin{align*}
\begin{array}{c}
\bullet
\end{array}
\end{align*}

\begin{align*}
\begin{array}{c}
\bullet
\end{array}
\end{align*}

\begin{align*}
\begin{array}{c}
\bullet
\end{array}
\end{align*}
Confluence reduction:
denoting a subset of the invisible transitions as confluent.

Non-probabilistically:
Non-probabilistic and probabilistic confluence reduction

Confluence reduction: denoting a \textit{subset of the invisible transitions} as confluent.

Non-probabilistically:

\begin{tikzpicture}
  \node (A) at (0,0) {a};
  \node (B) at (2,2) {a};
  \node (C) at (2,0) {b};
  \node (D) at (0,2) {c};
  \path [->] (A) edge node [right] {c} (B);
  \path [->] (A) edge node [right] {c} (C);
  \path [->] (B) edge node [right] {c} (D);
  \path [->] (C) edge node [right] {c} (D);
\end{tikzpicture}
Confluence reduction:
denoting a subset of the invisible transitions as confluent.

Non-probabilistically:
Confluence reduction:
denoting a subset of the invisible transitions as confluent.

Non-probabilistically:
Non-probabilistic and probabilistic confluence reduction

Confluence reduction:
\[\text{denoting a subset of the invisible transitions as confluent.}\]

Non-probabilistically:
\[
\begin{array}{c}
\bullet \\
\downarrow a \\
\bullet \\
\end{array} \xrightarrow{c} \begin{array}{c}
\bullet \\
\uparrow a \\
\bullet \\
\end{array}
\]

Probabilistically:
\[
\begin{array}{c}
\bullet \\
\downarrow a \\
\bullet \\
\end{array} \xrightarrow{c} \begin{array}{c}
\bullet \\
\uparrow a \\
\bullet \\
\end{array}
\]

\[
\begin{array}{c}
\bullet \\
\downarrow a \\
\bullet \\
\end{array} \xrightarrow{c} \begin{array}{c}
\bullet \\
\uparrow a \\
\bullet \\
\end{array}
\]

\[\text{equivalent}\]

Non-probabilistic and probabilistic confluence reduction

Confluence reduction:
denoting a subset of the invisible transitions as confluent.

Non-probabilistically:

Probabilistically:
Confluence reduction:
denoting a *subset of the invisible transitions* as confluent.

Non-probabilistically:

```
   a
↓   ↓
  ∘  ∘
  ↓   ↓
 b
```

Probabilistically:

```
   a
↓   ↓
  ∘  ∘
  ↓   ↓
 b
```
Partial Order Reduction:

- Preservation of \textit{probabilistic LTL}$\setminus \chi$
- Based on \textit{independent actions} and \textit{ample sets}
- Ample actions are allowed to be \textit{probabilistic}
Confluence versus Partial Order Reduction

Partial Order Reduction:

- Preservation of \( \text{probabilistic LTL} \setminus \chi \) (or \( \text{PCTL}^* \setminus \chi \))
- Based on independent actions and ample sets
- Ample actions are allowed to be probabilistic (or not)
Confluence versus Partial Order Reduction

Partial Order Reduction:
- Preservation of probabilistic $\text{LTL} \setminus \chi$ (or $\text{PCTL}^* \setminus \chi$)
- Based on independent actions and ample sets
- Ample actions are allowed to be probabilistic (or not)

Disadvantage: not defined for concrete state spaces
Confluence versus Partial Order Reduction

Partial Order Reduction:
- Preservation of \( \text{probabilistic LTL}\backslash\chi \) (or \( \text{PCTL}^*\backslash\chi \))
- Based on independent actions and ample sets
- Ample actions are allowed to be probabilistic (or not)

Disadvantage: not defined for concrete state spaces

Confluence Reduction:
- Preservation of \( \text{PCTL}^*\backslash\chi \)
- Based on confluent transitions (commuting diagrams)
Confluence versus Partial Order Reduction

Partial Order Reduction:
- Preservation of \( \text{probabilistic LTL} \backslash \chi \) (or \( \text{PCTL}^* \backslash \chi \))
- Based on independent actions and ample sets
- Ample actions are allowed to be probabilistic (or not)

Disadvantage: not defined for concrete state spaces

Confluence Reduction:
- Preservation of \( \text{PCTL}^* \backslash \chi \)
- Based on confluent transitions (commuting diagrams)

Disadvantage: confluent transitions not allowed to be probabilistic
Confluence versus Partial Order Reduction

Partial Order Reduction:
- Preservation of probabilistic LTL\(\chi\) (or PCTL*\(\chi\))
- Based on independent actions and ample sets
- Ample actions are allowed to be probabilistic (or not)

Disadvantage: not defined for concrete state spaces

Confluence Reduction:
- Preservation of PCTL*\(\chi\)
- Based on confluent transitions (commuting diagrams)

Disadvantage: confluent transitions not allowed to be probabilistic

Relating the notions:
- Confluence reduction subsumes branching-time POR
- Confluence reduction and linear-time POR are incomparable
Alterations to the concept of confluence

- Transitions may be mimicked by differently-labelled transitions
- Transitions only have to be invisible locally
- More liberal notion of equivalence of distributions
Alterations to the concept of confluence

- Transitions may be mimicked by *differently-labelled* transitions
- Transitions only have to be *invisible locally*
- More liberal notion of *equivalence of distributions*
Alterations to the concept of confluence

- Transitions may be mimicked by differently-labelled transitions
- Transitions only have to be invisible locally
- More liberal notion of equivalence of distributions
Alterations to the concept of confluence

- Transitions may be mimicked by differently-labelled transitions
- Transitions only have to be invisible locally
- More liberal notion of equivalence of distributions
Alterations to the concept of confluence

- Transitions may be mimicked by differently-labelled transitions
- Transitions only have to be invisible locally
- More liberal notion of equivalence of distributions
Alterations to the concept of confluence

- More liberal notion of equivalence of distributions

\[
\mu(S_i) = \nu(s_i) \land (S_i = \{s_i\} \lor \forall s \in S_i . \exists a \in \Sigma . s \xrightarrow{a} s_i \in T)
\]
Alterations to the concept of confluence

- More liberal notion of equivalence of distributions

\[
\mu(S_i) = \nu(s_i) \land (S_i = \{s_i\}) \lor \forall s \in S_i . \exists a \in \Sigma . s \xrightarrow{a} s_i \in T
\]
Alterations to the concept of confluence

- More liberal notion of equivalence of distributions

\[
\mu(S_i) = \nu(s_i) \land (S_i = \{s_i\} \lor \forall s \in S_i . \exists a \in \Sigma . s \xrightarrow{a} s_i \in T)
\]
Alterations to the concept of confluence

- More liberal notion of equivalence of distributions

\[
\mu(S_i) = \nu(s_i) \land (S_i = \{s_i\} \lor \forall s \in S_i \cdot \exists a \in \Sigma \cdot s \xrightarrow{a} s_i \in T)
\]

\[
\{(s, t) \mid s \in \text{spt}(\mu), t \in \text{spt}(\nu), s \xrightarrow{a} t \in T\}
\]
Correctness of confluence reduction

Even though
- Transitions may be mimicked by differently-labelled transitions
- Transitions only have to be invisible locally
- We have a more liberal notion of equivalence of distributions
Correctness of confluence reduction

Even though
  • Transitions may be mimicked by differently-labelled transitions
  • Transitions only have to be invisible locally
  • We have a more liberal notion of equivalence of distributions

Still we find:

**Theorem**

*Confluent transitions can be given priority, preserving $PCTL^*_X$.***
On-the-fly detection of confluence

Simulation using on-the-fly confluence detection:

1. Simulate until reaching a nondeterministic choice
On-the-fly detection of confluence

Simulation using on-the-fly confluence detection:

1. Simulate until reaching a nondeterministic choice
2. Check for each outgoing transition if it is confluent
   - If one choice is confluent, take it and continue
   - If no choice is confluent, abort
Simulation using on-the-fly confluence detection:

1. Simulate until reaching a nondeterministic choice
2. Check for each outgoing transition if it is confluent
   - If one choice is confluent, take it and continue
   - If no choice is confluent, abort

(Possibly, if confluence fails, attempt the same using POR)
Simulation using on-the-fly confluence detection:

1. Simulate until *reaching a nondeterministic choice*
2. Check for each outgoing transition if it is confluent
   - If one choice is confluent, take it and continue
   - If no choice is confluent, abort

(Possibly, if confluence fails, attempt the same using POR)

To check if a transition is confluent:

- Check if it is invisible
On-the-fly detection of confluence

Simulation using on-the-fly confluence detection:

1. Simulate until reaching a nondeterministic choice
2. Check for each outgoing transition if it is confluent
   - If one choice is confluent, take it and continue
   - If no choice is confluent, abort

(Possibly, if confluence fails, attempt the same using POR)

To check if a transition is confluent:

- Check if it is invisible
- Check if all its neighbouring transitions are mimicked
  - For this, additional transitions might need to be confluent
On-the-fly detection of confluence

Simulation using on-the-fly confluence detection:

1. Simulate until reaching a nondeterministic choice
2. Check for each outgoing transition if it is confluent
   - If one choice is confluent, take it and continue
   - If no choice is confluent, abort

(Possibly, if confluence fails, attempt the same using POR)

To check if a transition is confluent:

- Check if it is invisible
- Check if all its neighbouring transitions are mimicked
  - For this, additional transitions might need to be confluent

**Careful**: do not ignore visible behaviour forever (ignoring problem)
- Check if at least once every \( l \) steps a state is fully expanded
Checking a transition for confluence

\[ a \rightarrow b \]
\[ c \rightarrow d \]
\[ e \rightarrow f \]
\[ g \rightarrow h \]
\[ i \rightarrow j \]

Check if \( c \) is confluent
- No; it is not invisible
Check if \( a \) is confluent
- It is invisible
Is the \( c \)-transition mimicked?
- Possibly by the \( d \)-transition
- But then \( f \) has to be confluent: check this...
Checking a transition for confluence

- Check if \( c \) is confluent
Checking a transition for confluence

Check if $c$ is confluent
- No; it is not invisible
Checking a transition for confluence

- Check if $c$ is confluent
  - No; it is not invisible
- Check if $a$ is confluent
Checking a transition for confluence

Check if \( c \) is confluent
- No; it is not invisible

Check if \( a \) is confluent
- It is invisible
Checking a transition for confluence

- Check if $c$ is confluent
  - No; it is not invisible
- Check if $a$ is confluent
  - It is invisible
  - Is the $c$-transition mimicked?
Checking a transition for confluence

- Check if $c$ is confluent
  - No; it is not invisible
- Check if $a$ is confluent
  - It is invisible
  - Is the $c$-transition mimicked?
    - Possibly by the $d$-transition
Checking a transition for confluence

- Check if \( c \) is confluent
  - No; it is not invisible

- Check if \( a \) is confluent
  - It is invisible
  - Is the \( c \)-transition mimicked?
    - Possibly by the \( d \)-transition
    - But then \( f \) has to be confluent: check this
    - ...
Checking a transition for confluence

- Check if $c$ is confluent
  - No; it is not invisible
- Check if $a$ is confluent
  - It is invisible
  - Is the $c$-transition mimicked?
    - Possibly by the $d$-transition
    - But then $f$ has to be confluent: check this
    - ...
Checking a transition for confluence

- Check if c is confluent
  - No; it is not invisible
- Check if a is confluent
  - It is invisible
  - Is the c-transition mimicked?
    - Possibly by the d-transition
    - But then f has to be confluent: check this
    - . . .
Checking a transition for confluence

- Check if $c$ is confluent
  - No; it is not invisible
- Check if $a$ is confluent
  - It is invisible
  - Is the $c$-transition mimicked?
    - Possibly by the $d$-transition
    - But then $f$ has to be confluent: check this
    - ...
Checking a transition for confluence

- Check if $d$ is confluent
- Check if $b$ is confluent
- Is the $d$-transition mimicked?
- Possibly by the $e$-transition
- But then $g$ has to be confluent: check this
- It is not; abort
Checking a transition for confluence

- Check if $d$ is confluent
  - No; it is not invisible
Checking a transition for confluence

- Check if $d$ is confluent
  - No; it is not invisible
- Check if $b$ is confluent
Checking a transition for confluence

- Check if $d$ is confluent
  - No; it is not invisible
- Check if $b$ is confluent
  - It is invisible
Checking a transition for confluence

- Check if $d$ is confluent
  - No; it is not invisible
- Check if $b$ is confluent
  - It is invisible
  - Is the $d$-transition mimicked?
Checking a transition for confluence

- Check if \( d \) is confluent
  - No; it is not invisible
- Check if \( b \) is confluent
  - It is invisible
  - Is the \( d \)-transition mimicked?
    - Possibly by the \( e \)-transition
Checking a transition for confluence

- Check if $d$ is confluent
  - No; it is not invisible
- Check if $b$ is confluent
  - It is invisible
  - Is the $d$-transition mimicked?
    - Possibly by the $e$-transition
    - But then $g$ has to be confluent: check this
Checking a transition for confluence

- Check if $d$ is confluent
  - No; it is not invisible
- Check if $b$ is confluent
  - It is invisible
  - Is the $d$-transition mimicked?
    - Possibly by the $e$-transition
    - But then $g$ has to be confluent: check this
    - It is not; abort
Checking a transition for confluence

- Check if $d$ is confluent
  - No; it is not invisible
- Check if $b$ is confluent
  - It is invisible
  - Is the $d$-transition mimicked?
    - Possibly by the $e$-transition
    - But then $g$ has to be confluent: check this
    - It is not; **abort**
Tool Support

The Modest Toolset

Modest model

mctau TA
mcpta PTA
UPPAAL
PRISM

modes STA
prohver SHA
mod.PHAvEr

Results

modes: SMC for Modest/STA

⇒ MDP as special case
POR + Confluence

www.modestchecker.net

Arnd Hartmanns & Mark Timmer

On-the-fly Confluence Detection for SMC

NEW
Examples

**Dining Cryptographers**
N cryptographers, two neighbours each
Nondeterminism: communication order

**CSMA/CD** "DPTA"
Two senders, one shared channel, collisions
Nondeterministic choice of station inside channel

**BEB** (Bounded Exponential Backoff)
Detailed MDP model of exponential backoff
K: max. backoff, N: no of retries, H: no of hosts

---

Arnd Hartmanns & Mark Timmer
On-the-fly Confluence Detection for SMC
## Evaluation

### Results

<table>
<thead>
<tr>
<th>Model</th>
<th>Params</th>
<th>Uniform:</th>
<th>Partial Order:</th>
<th>Confluence:</th>
<th>Model Checking:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>time</td>
<td>time k s</td>
<td>time k s c t</td>
<td>states time</td>
</tr>
<tr>
<td>Dining Cryptographers</td>
<td>(3)</td>
<td>1 s</td>
<td>– – –</td>
<td>3 s 4 9 4.0 8.0</td>
<td>609 1 s</td>
</tr>
<tr>
<td></td>
<td>(4)</td>
<td>1 s</td>
<td>– – –</td>
<td>11 s 6 25 6.0 10.0</td>
<td>3 841 2 s</td>
</tr>
<tr>
<td></td>
<td>(5)</td>
<td>1 s</td>
<td>– – –</td>
<td>44 s 8 67 8.0 12.0</td>
<td>23 809 7 s</td>
</tr>
<tr>
<td></td>
<td>(6)</td>
<td>1 s</td>
<td>– – –</td>
<td>229 s 10 177 10.0 14.0</td>
<td>144 705 26 s</td>
</tr>
<tr>
<td></td>
<td>(7)</td>
<td>1 s</td>
<td>– – –</td>
<td>– timeout –</td>
<td>864 257 80 s</td>
</tr>
<tr>
<td>CSMA/CD</td>
<td>(2, 1)</td>
<td>2 s</td>
<td>– – –</td>
<td>4 s 3 46 5.4 16.4</td>
<td>15 283 11 s</td>
</tr>
<tr>
<td>(RF, BC&lt;sub&gt;max&lt;/sub&gt;)</td>
<td>(1, 1)</td>
<td>2 s</td>
<td>– – –</td>
<td>4 s 3 46 5.4 16.4</td>
<td>30 256 49 s</td>
</tr>
<tr>
<td></td>
<td>(2, 2)</td>
<td>2 s</td>
<td>– – –</td>
<td>10 s 3 150 5.1 16.0</td>
<td>98 533 52 s</td>
</tr>
<tr>
<td></td>
<td>(1, 2)</td>
<td>2 s</td>
<td>– – –</td>
<td>10 s 3 150 5.1 16.0</td>
<td>194 818 208 s</td>
</tr>
<tr>
<td>BEB</td>
<td>(4, 3, 3)</td>
<td>1 s</td>
<td>3 s 3 4</td>
<td>1 s 3 7 3.3 11.6</td>
<td>&gt; 10&lt;sup&gt;3&lt;/sup&gt; &gt; 0 s</td>
</tr>
<tr>
<td>(K, N, H)</td>
<td>(8, 7, 4)</td>
<td>2 s</td>
<td>7 s 4 8</td>
<td>4 s 4 15 5.6 16.7</td>
<td>&gt; 10&lt;sup&gt;7&lt;/sup&gt; &gt; 7 s</td>
</tr>
<tr>
<td></td>
<td>(16,15,5)</td>
<td>3 s</td>
<td>18 s 5 16</td>
<td>11 s 5 31 8.3 21.5</td>
<td>– memout –</td>
</tr>
<tr>
<td></td>
<td>(16,15,6)</td>
<td>3 s</td>
<td>40 s 6 32</td>
<td>34 s 6 63 11.2 26.2</td>
<td>– memout –</td>
</tr>
</tbody>
</table>

**+** Performance on BEB & CSMA/CD models vs. model-checking

**+** A bit faster than POR

**-** Does not work well for dining cryptographers

Arnd Hartmanns & Mark Timmer

On-the-fly Confluence Detection for SMC
A new approach to SMC for MDPs based on on-the-fly confluence detection

– detect confluence on-the-fly on the concrete state space
– handle more kinds of nondeterminism than POR method

<table>
<thead>
<tr>
<th>approach</th>
<th>nondeterminism</th>
<th>probabilities</th>
<th>memory</th>
<th>error bounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>POR</td>
<td>spurious interleavings</td>
<td>$P\downarrow \text{max} = P\downarrow \text{min}$</td>
<td>$s \ll n$</td>
<td>unchanged</td>
</tr>
<tr>
<td>conf. learning</td>
<td>confluent spurious</td>
<td>$P\downarrow \text{max} = P\downarrow \text{min}$</td>
<td>$s \ll n$</td>
<td>unchanged</td>
</tr>
<tr>
<td></td>
<td>any</td>
<td>$P\downarrow \text{max}$ only</td>
<td>$s \rightarrow n$</td>
<td>convergence</td>
</tr>
</tbody>
</table>

See also

www.modestchecker.net