Partial Order Reduction in Directed Model Checking

9th International SPIN Workshop

Grenoble
12.04.2002

Alberto Lluch Lafuente, Stefan Edelkamp, Stefan Leue
{lafuente,edelkamp,leue}@informatik.uni-freiburg.de.

Institut für Informatik
Albert-Ludwigs-Universität Freiburg
Germany
Combining Ample Set Partial Order Reduction and Heuristic Search for Finding LTL Safety Errors using Explicit-State Model Checking

9th International SPIN Workshop
Grenoble
12.04.2002

Alberto Lluch Lafuente, Stefan Edelkamp, Stefan Leue
{lafuente,edelkamp,leue}@informatik.uni-freiburg.de.

Institut für Informatik
Albert-Ludwigs-Universität Freiburg
Germany
Introduction

Directed Search → Partial Order → Partial Order + Directed Search

error state → ?

Partial Order Reduction in Directed Model Checking – p.3/15
Our Model Checking Framework

- Concurrent Asynchronous System: finite transition system.
- LTL$_x$ Safety Property.
- On-the-fly Model Checking.
- Safety error:
  - Path to error state in state transition graph.
Directed Model Checking

- Use guided, heuristic, directed search strategies for finding safety errors.

  - **Goals:**
    1. Reduce search effort.
    2. Provide short (meaningful) counterexamples.

- **Heuristic Search Strategy = Algorithm + Heuristics**
  - Algorithms: A*, Best-First, IDA*, …
  - Heuristics: based on specification, model, etc.
General State Expanding Search

State Space divided into three sets:

- $CLOSED$: visited and expanded states.
- $OPEN$: visited but not expanded states.
- $S \setminus (CLOSED \cup OPEN)$: not visited states.

In each step:
1. Extract a state $s$ from $OPEN$,
2. put its successors into $OPEN$,
3. move $s$ into $CLOSED$.

Examples: DFS ($OPEN$ as stack), BFS ($OPEN$ as queue).

Best-First: Extract states according to evaluation function.
Ample Set Method

Use $ample(s) \subseteq enabled(s)$ instead $enabled(s)$

Construct $M'$ such that
- $M'$ notably smaller than $M$
- $M'$ semantic equivalent to $M$

Semantic equivalence for $\text{LTL}_\neg X$: stuttering equivalence.

$\alpha, \beta$ are independent if $\forall s \in S$:
- they cannot disable each other.
- they are commutative.

$\alpha$ is invisible with respect to a set of propositions $P$ if $\forall s, s' \in S$ such that $s' = \alpha(s)$, $L(s) \cap P = L(s') \cap P$. 
Ample Set Construction

Four necessary and sufficient conditions for \textit{ample}:

**Condition C0**: \textit{ample}(s) is empty exactly when \textit{enabled}(s) is empty.

**Condition C1**: Along every path in the full state space that starts at \( s \), a transition that is dependent on a transition in \textit{ample}(s) does not occur without a transition in \textit{ample}(s) occurring first.

**Condition C2**: If a state \( s \) is not fully expanded, then each transition \( \alpha \) in the ample set must be invisible with regard to \( P \).

**Condition C3**: If for each state of a cycle in the reduced state space, a transition \( \alpha \) is enabled, then \( \alpha \) must be in the ample set of some of the states of the cycle.
Hierarchy of C3 Conditions

Depth-first search based algorithms

General state expanding algorithms

C3_{static}

C3_{stack} ← C3

C3_{duplicate} ← C3
**Reduction for Safety**

- **C3**: If for each state of a cycle in the reduced state space, a transition $\alpha$ is enabled, then $\alpha$ must be in the ample set of some of the successors of some of the states of the cycle.

- **Condition C3$_{stack}$**: If a state $s$ is not fully expanded, then at least one transition in $ample(s)$ does not lead to a state on the search stack.

- **C3$_{duplicate}$**: If a state $s$ is not fully expanded, then at least one transition in $ample(s)$ does not lead to an already visited state.

- **Condition C3$_{static}$** cannot be simplified.
PO and Counterexample Length

- Shortest path to an error state in the reduced state space may be longer than the shortest path in the full one.

- Example: $\phi = \Box p$, $\alpha, \beta$ independent, $M \sim_{st} M'$.
Eliminate *irrelevant* transitions from counterexample trace.

A transition is irrelevant if:
1. it is independent on each relevant transition occurring after it,
2. and it is invisible.

The resulting trace:
1. Corresponds to an execution in the system.
2. Is stuttering equivalent to the original one.
Some Experimental Results

- Better $\mathbf{C3}$ condition for stack-based algorithms?
  - $\mathbf{C3}_{\text{stack}}$ (as expected).

- Better $\mathbf{C3}$ condition GSEA (A*,BF)?
  - $\mathbf{C3}_{\text{static}}$ in most cases, but not always.

- Counterexample length in reduced space?
  - Phenomenon observed in one case only.
  - Proposed method shortens counterexample.
More Experimental Results

- Combined Reduction Effect?
- Reduction in the number of states: Separate vs. Combined.

<table>
<thead>
<tr>
<th>model</th>
<th>H</th>
<th>PO</th>
<th>H*PO</th>
<th>H+PO</th>
</tr>
</thead>
<tbody>
<tr>
<td>leader</td>
<td>2.3</td>
<td>40.8</td>
<td>93.8</td>
<td>267.0</td>
</tr>
<tr>
<td>pots</td>
<td>5.9</td>
<td>1.4</td>
<td>8.26</td>
<td>9.5</td>
</tr>
<tr>
<td>marriers</td>
<td>1.9</td>
<td>2.7</td>
<td>5.1</td>
<td>5.9</td>
</tr>
<tr>
<td>giop</td>
<td>1.3</td>
<td>2.6</td>
<td>3.4</td>
<td>3.3</td>
</tr>
</tbody>
</table>

- Synergy in some cases.
Conclusion

- Condition \( C_3 \) depends on the algorithm:
  - \( C_{3_{\text{stack}}} \) for stack-based strategies (DFS, IDA*).
  - \( C_{3_{\text{duplicate}}} \) and \( C_{3_{\text{sticky}}} \) for GSEA.

- Partial Order can lead to larger counterexamples:
  - Possible solution: eliminate irrelevant transitions.

- Combined reduction effect:
  - Synergy in some cases.