Introducing LoLA 2.0

Explicit State Space Exploration for Petri Nets

Karsten Wolf, University of Rostock
# Commercial I

## Functional Verification of Task Partitioning for Multiprocessor Embedded Systems

DIPANKAR DAS, P. P. CHAKRABARTI, and RAJEEV KUMAR

Indian Institute of Technology Kharagpur

<table>
<thead>
<tr>
<th></th>
<th>nodes UML</th>
<th>places Petri Net</th>
<th>transitions Petri Net</th>
<th>Spin</th>
<th>LoLA</th>
<th>PROD</th>
<th>PEP</th>
<th>TraceMatch</th>
</tr>
</thead>
<tbody>
<tr>
<td>crc32_T</td>
<td>70</td>
<td>90</td>
<td>30</td>
<td>580 ms</td>
<td>10 ms</td>
<td>2170 ms</td>
<td>30 ms</td>
<td>1 ms</td>
</tr>
<tr>
<td>crc32_F</td>
<td>70</td>
<td>90</td>
<td>30</td>
<td>600 ms</td>
<td>10 ms</td>
<td>2200 ms</td>
<td>30 ms</td>
<td>2 ms</td>
</tr>
<tr>
<td>adpcm_T</td>
<td>85</td>
<td>102</td>
<td>66</td>
<td>630 ms</td>
<td>10 ms</td>
<td>2340 ms</td>
<td>20 ms</td>
<td>2 ms</td>
</tr>
<tr>
<td>adpcm_F</td>
<td>85</td>
<td>102</td>
<td>66</td>
<td>590 ms</td>
<td>10 ms</td>
<td>2420 ms</td>
<td>20 ms</td>
<td>2 ms</td>
</tr>
<tr>
<td>fft_T</td>
<td>98</td>
<td>76</td>
<td>106</td>
<td>600 ms</td>
<td>&lt;10 ms</td>
<td>2520 ms</td>
<td>20 ms</td>
<td>1 ms</td>
</tr>
<tr>
<td>fft_F</td>
<td>98</td>
<td>76</td>
<td>106</td>
<td>640 ms</td>
<td>20 ms</td>
<td>2660 ms</td>
<td>20 ms</td>
<td>3 ms</td>
</tr>
<tr>
<td>basicmath_T</td>
<td>265</td>
<td>320</td>
<td>157</td>
<td>1700 ms</td>
<td>20 ms</td>
<td>11300 ms</td>
<td>170 ms</td>
<td>9 ms</td>
</tr>
<tr>
<td>basicmath_F</td>
<td>265</td>
<td>320</td>
<td>157</td>
<td>1880 ms</td>
<td>40 ms</td>
<td>12420 ms</td>
<td>180 ms</td>
<td>27 ms</td>
</tr>
<tr>
<td>dijkstra_T</td>
<td>285</td>
<td>302</td>
<td>229</td>
<td>1340 ms</td>
<td>20 ms</td>
<td>8570 ms</td>
<td>140 ms</td>
<td>23 ms</td>
</tr>
<tr>
<td>dijkstra_F</td>
<td>285</td>
<td>302</td>
<td>229</td>
<td>1330 ms</td>
<td>80 ms</td>
<td>8750 ms</td>
<td>140 ms</td>
<td>27 ms</td>
</tr>
<tr>
<td>sha_T</td>
<td>309</td>
<td>314</td>
<td>239</td>
<td>2050 ms</td>
<td>20 ms</td>
<td>13640 ms</td>
<td>200 ms</td>
<td>8 ms</td>
</tr>
<tr>
<td>sha_F</td>
<td>309</td>
<td>314</td>
<td>239</td>
<td>2020 ms</td>
<td>50 ms</td>
<td>13550 ms</td>
<td>210 ms</td>
<td>15 ms</td>
</tr>
</tbody>
</table>
The Pathway Logic Assistant

Carolyn Talcott\textsuperscript{1} and David L. Dill\textsuperscript{2}

\textsuperscript{1} SRI International
clt@csl.sri.com
\textsuperscript{2} Stanford University
dill@cs.stanford.edu

Figure 1 shows the result of the Petri net query corresponding to prop3 found by the LoLA [13] Petri net analysis tool, as displayed by PLA using the Pathalyzer tool. Lola uses “stubborn set reduction”, which is a technique that exploits the ease of determining the independence of certain transitions in the Petri nets. For reachability queries on our nets, answering a reachability query that would have taken hours using a general purpose model-checking tool takes on the order of a second in LoLA—fast enough to permit interactive use.
Commercial III

Experimental Results

Checking coverability on standard Petri Net benchmarks:

Comparison with numerous algorithms implemented in MIST
# Commercial IV

## Model Checking Contest @ Petri Nets 2013

*Milano, Italy, June 25, 2013*

---

### Introduction

This page summarizes the trophies won by the tools for *ReachabilityMix*. At this stage, we think this is not possible to provide a global trophy for the MCC @ Petri Net 2013.

#### Separated trophies for ReachabilityMix

We first consider separately the two types of models.

<table>
<thead>
<tr>
<th>Trophies for the &quot;Known&quot; Models</th>
<th>Trophies for the &quot;Surprise&quot; Models</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="LoLA optimistic 189 (points)" /></td>
<td><img src="image2" alt="LoLA optimistic 154 (points)" /></td>
</tr>
<tr>
<td><img src="image3" alt="LoLA 184 (points)" /></td>
<td><img src="image4" alt="Marcie 24 (points)" /></td>
</tr>
<tr>
<td><img src="image5" alt="LoLA optimistic incomplete 12 (points)" /></td>
<td><img src="image6" alt="LoLA 12 (points)" /></td>
</tr>
<tr>
<td><img src="image7" alt="LoLA optimistic incomplete 12 (points)" /></td>
<td></td>
</tr>
</tbody>
</table>

#### Global trophies for ReachabilityMix

For this trophy, we use the following formula: results on "Known" models *+ 2 x results on "Surprise" models.*

<table>
<thead>
<tr>
<th>Trophies for All Models</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image8" alt="LoLA optimistic 213 (points)" /></td>
</tr>
<tr>
<td><img src="image9" alt="LoLA 208 (points)" /></td>
</tr>
<tr>
<td><img src="image10" alt="Marcie 166 (points)" /></td>
</tr>
<tr>
<td><img src="image11" alt="LoLA optimistic incomplete 166 (points)" /></td>
</tr>
</tbody>
</table>
Petri Net = Vector Addition System

\[ m_0: (4, 0, 0, 1, 0) \]

\[ t_2 = (-1, 1, 1, -1, 0) \]

If \( m_0 \xrightarrow{t_2} m_1 \) then \( m_0 + t_2 = m_1 = (3, 1, 1, 0, 0) \)
Incidence Matrix

\[
C = \begin{pmatrix}
1 & -1 & 0 & 0 & 0 \\
-1 & 1 & 0 & 0 & 0 \\
0 & 1 & -1 & 0 & 1 \\
0 & -1 & 0 & 1 & 0 \\
0 & 0 & 1 & -1 & -1
\end{pmatrix}
\]
What is LoLA?

• State space exploration tool for Petri nets
• Since 1997
• Tuned to efficiency
• No graphical interface
• Many reduction techniques + combinations
  – Symmetry
  – Partial Order
  – Sweep-Line
  – Coverability
  – ...
• Specialized to simple properties:
  – Reachability ... EF φ
  – Deadlock
  – Liveness ... AG EF φ
  – Boundedness ...exists k: AG p ≤ k
  – Dead transitions AG NOT FIREABLE(t)
  – ...

EF φ – Dead transitions AG NOT FIREABLE(t)
From 1.x to 2.0

• Complete Re-Implementation
  – Clear modular structure
  – State-of-the-art software engineering
  – Several stud

• LTL model checking added

• Unified property specification in CTL*

• Single executable

• Multicore support
LoLA 2.0 Architecture
LoLA 2.0 Architecture

- Depth first
- Breadth first
- Sweep-Line
- Coverability
- Findpath
- TSCC
- LTL
- CTL

Planning

- Property
- Net
- Exploration
- Firelist
- Encoder
- Witness
- Reporting
- Store
The Sweep-Line-Method

Idea: additional information:
State $s \rightarrow \text{Progress-Value } p(s)$,

Such that: $s \rightarrow s'$ implies $p(s) \leq p(s')$

Effect:
- Unprocessed
- Not yet seen
- Processed

Lola:
- Only serious implementation
- Can be combined with other techniques
- $P$ determined automatically
Coverability graph

Overapproximation of state space:

(1, 0, 0, 2, 3)
(1, 1, 0, 3, 3)
(1, 0, 0, 5, 3)
(1, 4, 0, 5, 3)
(1, 3, 0, 9, 3)
(1, 8, 0, 11, 3)

→

(1, oo, 0, oo, 3)

- Always finite

Lola:
- Only few serious implementations
- Can be combined with other techniques
- Partial evaluation of nontrivial CTL properties
LoLA 2.0 Architecture

Copy
Significant
Compressed
Variable-length
Symmetry

Planning

Property
Encoder
Store

Net
Exploration
Witness

Firelist
Reporting
Encoders

• Significant:

  Use place invariants:

  \[ fo[i] + hl[i] + ea[i] + ea[i-1] + hr[i-1] = \text{const}. \]

  \( \rightarrow \) do not store \( fo[i] \)

• Compressed

  \( (3,0,4,1,0,2) \)

  \( \rightarrow \)

  \( (11001001010) \)

  10 phils = 50 places = 7 bytes per state
Symmetry

Only store/explore one representative

Lola:
- Automated detection (graph automorphisms)
- Can be combined
LoLA 2.0 Architecture

- Property
- Net
- Firelist
- Exploration
- Encoder
- Witness
- Reporting
- Store

Prefix-Tree
Bloom-Filter
Cycle
Bloom-Filter

- Hash: States $\rightarrow$ Nat
- Only mark used hash values
- Improved coverage by using several independent hash functions

Lola:
- Can choose number of hash functions
- Can be combined with other techniques
Cycle store

- Store only some states
- Take care that every cycle has a stored state

Lola:
- States to be stored determined from incidence matrix
LoLA 2.0 Architecture

- Property
- Exploration
- Net
- Firelist
- Encoder
- Witness
- Reporting

Lola:
- Special support for simple properties

Deadlock, Reachability, TSCC, CTL, LTL
TSCC-based properties

- AG EF $\phi$:
- EF AG $\phi$:
- EF AG EF $\phi$:
- AG EF AG $\phi$:

TSCC can be detected using only one stack
Many properties can be *made* simple

*There is a path from start to end that includes an occurrence of t:*

\[ EF \ (t \text{ and } EF \text{ end}) \]

*Or*

\[ EF \text{ end and } p=0 \]
Brute force
*Stubborn sets*

**LoLA 2.0 Architecture**

- Planning
  - Firelist
  - Exploration
  - Net
  - Property
- Encoder
- Witness
- Reporting
- Store
Stubborn sets

stubborn(s): subset of transitions in s

→ reduced transition system:
  fire only transitions in stubborn(s)

Choose function stubborn such that property is preserved
Comparison

Others:
• Preserve LTL, CTL, ...

• Priority on invisible transitions

LoLA:
• Preserve Reachability, AGEF, ....

• Priority on progress
PoLA 2.0 Architecture

Path
Partially ordered run
highlights

Planning
- Property
- Net
- Exploration
- Firelist
- Witness
- Encoder
- Reporting
- Store
LoLA 2.0 Architecture

- **To screen**
  - **To File (JSON)**
  - **To UDP**

**Planning**
- Property
- Exploration
- Net
- Firelist
- Encoder
- Witness
- Reporting
- Store
Analyse property & select suitable modules

HoLA 2.0 Architecture

Planning

- Property
- Net
- Exploration
- Firelist
- Encoder
- Witness
- Reporting
- Store
Success Stories

- Asynchronous circuits: reachability
- Workflow models: liveness, boundedness
- Web service composition by planning: reachability
- Biochemical reaction chains: reachability
- Cutoffs in parameterized verification: coverability
- Multiprocessor embedded systems: deadlock
How to use LoLA

• Make property simple
• Many simple queries are better than few complicated ones
• Embed in scripts, graphical tools, LoLA as a service,...
• Get Petri nets by translation
  – Pathway logic
  – UML-ACT
  – WS-BPEL
  – PLANICS
  – ...

Conclusion

- simple
- Sweep-line coverability
- symmetry
- Bloom cycle
- combination

Planning
- Property
- Encoder
- Witness
- Reporting
- Store

Exploration
- Net
- Firelist
- Stubborn set