THE EFFICIENT SYMBOLIC TOOLS PACKAGE

Robert Meolic, Tatjana Kapus, Zmago Brezočnik
Faculty of Electrical Engineering and Computer Science
University of Maribor
Smetanova ul. 17, SI-2000 Maribor, Slovenia
{meolic,kapus,brezocnik}@uni-mb.si

Contents:
1. Introduction
2. Simple process algebra
3. Symbolic verification
4. EST - Efficient Symbolic Tools
5. An example of verification
6. Conclusion
THE EFFICIENT SYMBOLIC TOOLS PACKAGE

Introduction

- **Process algebras** are widely used formalisms in the verification of concurrent systems.
- A process is an entity which **performs actions** and **transits** between its internal states.
- Processes **synchronise** with each other by simultaneously performing synchronisation actions.

**Milner’s CCS** (Calculus of Communicating Systems, 1980)

\[ Z = \tau \cdot (\alpha(x) \cdot \delta(0) + \beta(x) \cdot \delta(1)) \cdot Z \]

**Hoare’s CSP** (Communicating Sequential Processes, 1985)

\[ T = left ? x \rightarrow right ! 0 \rightarrow right ! 1 \rightarrow T \]
Simple process algebra

PROCESS Drink_Machine
INITIAL STATE s1
TRANSITIONS s1 = coin?.s2
    s2 = tea!.s3
    s2 = coffee!.s4

PROCESS User
INITIAL STATE s1
TRANSITIONS s1 = coin!.s2
    s2 = coffee?.s3
Parallel composition

AUTOMAT

USER

Parallel composition
Symbolic verification

- States and transitions are encoded by Boolean functions rather than being explicitly enumerated.
- Further, Boolean functions are represented with binary decision diagrams (BDDs).
- Thus, operations on processes are performed as operations with Boolean functions, which are actually performed as manipulations with BDDs.
Verification Methods

• By equivalence testing:
  – trace equivalence,
  – observational equivalences,
  – testing equivalence.

• By model checking:
  – with ACTL,
  – with $\mu$-calculus.
EST - Efficient Symbolic Tools

- EST is a new tool for the verification of concurrent systems, which has not been widely presented yet.
- EST is a relatively small package written in C with a Tcl/Tk user interface.
- Main advantages: flexibility, portability and an efficient memory management.

EST is a modularized package:

- **Binary Decision Diagrams** module is a general purpose BDD package for the manipulation of Boolean functions,
- **Process Algebra** module is a framework for representing processes,
- **Versis** module implements operations on processes,
- **Model Checking** module provides functions for ACTL model checking,
- **My Interface** module implements the user interface.
An overview of EST
An example of verification

Milner’s simple distributed scheduler consists of

- starter $S$,
- cyclers $C_i$.

External behaviour of the system with $k$ cyclers
The results of verification

- HP 715/100 with 128 MB RAM,
- The program was allowed to have at most 500000 BDD nodes at once, so that the total memory consumption never exceeded 32 MB.
- 7 times better results on an overclocked Pentium II 266 with Linux.

<table>
<thead>
<tr>
<th>$k$</th>
<th>states</th>
<th>transitions (without $\tau$)</th>
<th>nodes in BDD</th>
<th>parallel composition</th>
<th>weak obs. equivalence</th>
<th>testing equivalence</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>97</td>
<td>241 (32)</td>
<td>272</td>
<td>0.5s</td>
<td>0.3s</td>
<td>0.4s</td>
</tr>
<tr>
<td>8</td>
<td>3073</td>
<td>13825 (1024)</td>
<td>707</td>
<td>2.1s</td>
<td>0.8s</td>
<td>1.3s</td>
</tr>
<tr>
<td>12</td>
<td>73729</td>
<td>479233 (24576)</td>
<td>1245</td>
<td>7.1s</td>
<td>1.6s</td>
<td>2.9s</td>
</tr>
<tr>
<td>16</td>
<td>1572865</td>
<td>13369345 (524288)</td>
<td>1912</td>
<td>19.5s</td>
<td>2.8s</td>
<td>4.9s</td>
</tr>
<tr>
<td>20</td>
<td>31457281</td>
<td>(10485760)</td>
<td>2674</td>
<td>47.5s</td>
<td>4.1s</td>
<td>8.2s</td>
</tr>
<tr>
<td>24</td>
<td>603979777</td>
<td>-</td>
<td>3548</td>
<td>97.8s</td>
<td>5.8s</td>
<td>12.8s</td>
</tr>
<tr>
<td>28</td>
<td>-</td>
<td>-</td>
<td>4534</td>
<td>205.1s</td>
<td>8.0s</td>
<td>18.7s</td>
</tr>
<tr>
<td>32</td>
<td>-</td>
<td>-</td>
<td>5665</td>
<td>452.8s</td>
<td>10.6s</td>
<td>26.9s</td>
</tr>
</tbody>
</table>
Conclusion

• EST project started in 1992,
  see http://www.el.feri.uni-mb.si/est/

• EST distinguishes itself as a small and efficient package with an easily readable source code and well implemented algorithms.

• EST uses symbolic methods to represent and manipulate processes.

• EST has already been successfully used for the verification of some larger concurrent systems, for example the problem of simple crossing of a road and a railway and formal verification of bounded retransmission protocol.

• The future work:
  – diagnostic (witnesses, counterexamples),
  – introduction of explicit data-passing.