June 22nd 2270

Efficient BDD Representation for Reachability Analysis of Timed Automata

Dirk Beyer

Software Systems Engineering Research Group
BTU Cottbus
www.software-systemtechnik.de/Rabbit
Problems of existing approaches

• Modeling
  – flat set of communicating automata
  – structure of system is lost in model
    ➔ confusing, hard to understand, not modular

• Verification
  – explicit enumeration of discrete states, exponential
  – clock valuations are represented by matrices
  – non-convex sets require more than one matrices
  – CDDs are used, but without regarding variable ordering
    ➔ inefficient, exponential effort

• Case studies
  – scalable benchmark examples with regular structure
  – few models with realistic, unregular structure
    ➔ existing case studies to small

Example: Fischer’s protocol

Uncritical

Assign

Critical

Wait

Timed Automata for process i

\[
k = 0 \\
x_i = c
\]

\[
k = 0 \\
x_i' = 0
\]

\[
k = 0 \\
x_i' = 0
\]

\[
x_i' = 0
\]

\[
k' = 0
\]

\[
x_i \geq c \\
k' = 0
\]

\[
k' = i
\]

\[
x_i' = 0
\]

\[
x_i \geq c \\
k \neq i
\]

\[
x_i \geq c \\
k = i
\]
Modularity in Cottbus Timed Automata

Module RailRoadCrossing:
LOCAL
lower : SYNC
raise : Sync
app  : Sync
exit : SYNC

Module Controller:
INPUT
app   : Sync
exit  : Sync
OUTPUT
lower : Sync
raise : Sync
LOCAL
! : CLOCK

Module Environment:
INPUT
lower : Sync
raise : Sync
app   : Sync
exit  : Sync

Module Gate:
INPUT
lower : Sync
raise : Sync
LOCAL
g    : Analog

Module Train:
OUTPUT
app   : Sync
exit  : Sync

Hierarchy

© Software Systems Engineering Research Group, BTU Cottbus, 2002
Reuse of components

Refinement steps for large systems

Abstract system

Refinement

Detailed system

Refinement

© Software Systems Engineering Research Group, BTU Cottbus, 2002
Problems of existing approaches

• Modeling
  – hierarchical model, abstraction layers
  – reuse of components, substitution of modules
  ➔ structured, understandable, maintainable, because modular, useful for verification

• Verification
  – explicit enumeration of discrete states, exponential
  – clock valuations are represented by matrices
  – non-convex sets require more than one matrices
  – CDDs are used, but without regarding variable ordering
  ➔ inefficient, exponential effort

• Case studies
  – scalable benchmark examples with regular structure
  – few models with realistic, unregular structure
  ➔ existing case studies to small

Verification of real-time systems

• Formalism for modular modeling
  – Theoretical basis: timed and hybrid automata
  – Module concept
  ➔ Good for modeling large systems

• Reachability analysis using BDD representation
  – Integer semantics
  – Estimate-based variable ordering
  ➔ Very efficient

• Refinement checking
  – Simulation relation
  ➔ Modular Proofs
Reachability analysis

- Verification of safety properties
  - Performance problems with existing tools
- 1st problem: explicit discrete states
  - BDD representation (own package)
- 2nd problem: separated clock representation
  - discrete TA-semantics, using also BDDs
- 3rd problem: variable orderings
  - heuristic using communication structure

Contribution to efficiency

- Evaluation of our method for three examples (Fischer, FDDI, CSMA/CD)
- 1. there exists a variable ordering for polynomial size complexity
- 2. our tool is able to find such variable ordering automatically
- 3. empirical evidence

- industrial case study: the approach works
- the approach is applicable for DDs in general
### Complexity results

<table>
<thead>
<tr>
<th>Protocol</th>
<th>BDD size</th>
<th>CDD size (location-first)</th>
<th>CDD size (smallest)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fischer</td>
<td>$\Theta(c^3 n^2 \lg c)$</td>
<td>$\Omega(2^n)$</td>
<td>$\Theta(n^3)$</td>
</tr>
<tr>
<td>CSMA/CD</td>
<td>$\Theta(n^3 \lg \lambda)$</td>
<td>$\Omega(3^n)$</td>
<td>$\Theta(n^2)$</td>
</tr>
<tr>
<td>Token ring</td>
<td>$\Theta(n^2 ttr t \lg ttr)$</td>
<td>$\Theta(n^2)$</td>
<td>$\Theta(n^2)$</td>
</tr>
<tr>
<td>FDDI</td>
<td>= $\Theta(n^4 \lg n)$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Complexity analysis: Fischer's protocol

![Timed Automata for process i](image)

- **Uncritical**
  - $k = 0$
  - $x_i = c$
  - $x_i' = 0$
  - $k' = 0$

- **Critical**
  - $x_i \geq c$
  - $k \neq i$

- **Assign**
  - $x_i \leq c - 1$
  - $x_i' = 0$

- **Wait**
  - $k' = i$
  - $x_i' = 0$

- **Timed Automata for process i**

© Software Systems Engineering Research Group, BTU Cottbus, 2002
Communication graph

- General characteristics of „good“ variable orderings:
  - Communicating components have neighboring positions
  - Components which communicate with many other components at first

- Automatic ordering:
  - Estimation for the size of the BDD evaluates different variable orderings

BDD structure and size estimation

\[ \sum_{i=1}^{n} \left( 2^{|q_i|} - 1 \right) \cdot \left( \prod_{k \in \text{Comm}_A(i)} |Q_k| \right) \]
Example: Fischer's mutex protocol

TA for process i:

Communication graph:

Variable k

Auto. 1

Clock 1

Auto. 2

Clock 2

Auto. n

Clock n

Different variable orderings

Consider 3 different variable orderings for the BDD representing the reachable set of Fischer’s protocol:

Separated

k at end

k in front

© Software Systems Engineering Research Group, BTU Cottbus, 2002
Measurement: Fischer's protocol

Comparison with other approaches
BDD for the reachable set

BDDs for the reachable sets
BDDs for the reachable sets

k in front versus k at end

AND 4

Estimate vs. real size

estimated BDD

real BDD
Problems of existing approaches

• Modeling ✔
  – hierarchical model, abstraction layers
  – reuse of components, substitution of modules
  ➔ structured, understandable, maintainable, because modular

• Verification ✔
  – unique symbolic representation of states AND clock valuations
  – proved integer semantics, proved upper bound for trans rel
  – method for complexity analysis, polynomial is possible
  – method for variable ordering, good orderings can be computed
  ➔ efficient, polynomial effort for some examples

• Case studies
  – scalable benchmark examples with regular structure
  – few models with realistic, irregular structure
  ➔ existing case studies to small
Complete production line example (1)

ordering derived from modular structure

Complete production line example (2)

ordering derived from modular structure
ordering based on calculated size estimates
Problems of existing approaches

• **Modeling**✓
  - hierarchical model, abstraction layers
  - reuse of components, substitution of modules
  ➞ structured, understandable, maintainable, because modular

• **Verification**✓
  - unique symbolic representation of states AND clock valuations
  - proved integer semantics, proved upper bound for trans rel
  - method for complexity analysis, polynomial is possible
  - method for variable ordering, good orderings can be computed
  ➞ efficient, polynomial effort for some examples

• **Case studies**✓
  - scalable benchmark examples => polynomial effort
  - large production cell model => verifiable using modular structure
  ➞ approach works, is relevant for practice, executable controller can be synthesized from proven model

Results

• **Modular formalism for modelling**

• **Efficient verification using BDD representation**
  - Complexity analysis of reach sets: good ordering exists
  - Good variable orderings can be found by tool using modular structure of model

• **Tool framework for timed and hybrid automata**
  - Double Description Method for the hybrid case
  - Binary Decision Diagrams for the timed case

• **Different verification strategies for modular proofs**
  - Reachability analysis for safety properties
  - Refinement check via simulation relation

• **Several case studies:**
  - AND circuit, Fischer’s protocol, CSMA/CD, FDDI,
  - Production cell, controller synthesis
FDDI protocol

CSMA/CD protocol