Integrating Simulink into the Model-Checker Cosmos

Benoît Barbot\(^{(a)}\), Béatrice Bérard\(^{(b)}\), Yann Duplouy\(^{(c,d)}\), Serge Haddad\(^{(d)}\)

\(^{(a)}\) LACL, Université Paris-Est Créteil
\(^{(b)}\) Sorbonne Université, LIP6, CNRS UMR 7606, Paris, France
\(^{(c)}\) IRT SystemX, Paris-Saclay, France
\(^{(d)}\) LSV, ENS Paris-Saclay, CNRS, Inria, Université Paris-Saclay

MeFoSyLoMa: June, 15th 2018

at LIPN, Université Paris 13
Description [Ballarini, Barbot, Duflot, Haddad, Pekergin 2015]

- Statistical model-checker for HASL over stochastic nets;
- Free software (GPLv3); C++, OCaml;

Main Applications

- Flexible manufacturing systems;
- Biological networks [Barbot, Kwiatkowska 2015];
- Embedded pacemaker model [Barbot, Kwiatkowska, Mereacre, Paoletti 2015].
Simulink®

**Description**
- Block diagram modeling and simulation for hybrid systems;
- Commercial software, embedded into MathWork’s MATLAB.

**Applications**
- Advanced driver-assistance systems;
- Signal processing, etc.
Motivations

- Modeling cyber-physical systems: combining probabilistic features and differential equations;
- Providing semantics for Simulink models which increases the confidence in results;
- Improving efficiency;
- Lessening the dependency on MathWorks.
<table>
<thead>
<tr>
<th>1</th>
<th>Semantics for Simulink</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Integrating Simulink into Cosmos</td>
</tr>
<tr>
<td>3</td>
<td>Benchmarks</td>
</tr>
</tbody>
</table>
Outline

1. Semantics for Simulink
2. Integrating Simulink into Cosmos
3. Benchmarks
Challenges

Increasing the confidence

- Documentation is mostly informal;
- Development of models on what you think you know;

⇒ Necessity of a formal semantics.

Difficulties of such a formalisation

- Implementation of Integration
- Research of thresholds
- Managing blocks with delay

⇒ Necessity of an approximate formal semantics.
Overview of Simulink Syntax

- Simulation Interval: $\text{Time} = [t_0, t_{\text{end}}]$
- Signals: $\text{Time} \rightarrow \mathbb{R}$ right-continuous;
- Characteristics of blocks:
  - continuous or discrete-time evaluation;
  - threshold crossings;
  - possible latency;

Simulation Equation:

$$0 \quad B_1 \quad \begin{array}{c}
\dot{y} \\
1
\end{array} \quad B_2 \quad \frac{1}{s} \quad y \quad B_3 \quad \begin{array}{c}
\dot{z} \\
1
\end{array} \quad B_4 \quad z$$

- $B_1$: \( \text{init}_2 = 0 \)
- $B_2$: \( r_3 = 1 \)
- $B_3$: \( \text{init}_4 = 0 \)
- $B_4$: \( \text{init}_3 = 0 \)
Providing an exact semantics

Trajectory: vector $\vec{w}$ of all values of output signals.

Establishing differential equation systems

- *modes* for threshold crossings;
- *history* for blocks with latency.

Splitting the simulation interval: $\text{Time} = \bigcup_{i=0}^{N-1} [t_i, t_{i+1}]$

- no sampling inside an interval $[t_i, t_{i+1}]$;
- positive latencies exceed $t_{i+1} - t_i$;
- "consistent" changes of modes between intervals.
Providing an exact semantics – Example

\[
\begin{align*}
\dot{y}(t) &= 1 \\
\dot{z}(t) &= 0 \quad \Rightarrow \quad \begin{cases} y(t) = t \\ z(t) = 0 \end{cases}
\end{align*}
\]

Over \([0, 1]\)

\[
\begin{align*}
\dot{y}(t) &= z(t) \\
\dot{z}(t) &= t - 1 \quad \Rightarrow \quad \begin{cases} y(t) = (t - 1)^3/6 + 1 \\ z(t) = (t - 1)^2/2 \end{cases}
\end{align*}
\]

Time = \([0, 2] = [0, 1] \cup [1, 2]\)

Over \([0, 1]\)

\[
\begin{align*}
\dot{y}(t) &= 1 \\
\dot{z}(t) &= 0 \quad \Rightarrow \quad \begin{cases} y(t) = t \\ z(t) = 0 \end{cases}
\end{align*}
\]

Over \([1, 2]\)
Accurate approximate semantics

**Problem:** Exact semantics is not operational;

**Goal:** Approximation of $\vec{w}$

**Principle:** Iterative construction of the sub-interval partition

$$\text{Time} = \bigcup_{i=0}^{N-1} [t_i, t_{i+1}].$$

- Signal values stored at each $t_i$: array $W_B, o[i]$.
  Linear interpolation for intermediate values.

- Next evaluation time:
  - Adaptative Integration Step: Runge-Kutta-Fehlberg (ODE45)
  - Threshold crossing: interpolation of candidate time instants.
Outline

1. Semantics for Simulink
2. Integrating Simulink into Cosmos
3. Benchmarks
Challenges

Different kinds of simulation

- **COSMOS** (until now): simulation of stochastic discrete-event systems.
- Simulink: formalism (and tool) for continuous variable dynamic systems.

$\Rightarrow$ Extend **COSMOS** to simulate Simulink models communicating with stochastic systems.

Communication between models

- How to transmit information?
- How to schedule simulation events?
**Communication**

**Interface transition for (Simulink) inputs**
- Input place modification triggers firing;
- New content is read;
- C code computes output signals.

**Interface transition for (Simulink) outputs**
- Output places are connected by read arcs;
- Simulink step triggers firing;
- C code rewrites output places.
Principles of discrete-event simulation

Events
An event is composed of ID, absolute time, priority and tie.

Event Queue
Events are accordingly ordered in a queue (usual implementation: heap).

Simulation Loop
- Extract earliest event from event queue;
- Update state due to the event occurrence;
- Remove disabled events from queue;
- Add to the queue newly enabled events.

Observation. Simulink being sequential, it has a single event.
Simulation Loop for Cosmos/Simulink

Depending on the extracted event:

1. If it is a net transition firing:
   - Update marking;
   - Execute C code;
   - Generate new net events possibly including Simulink input transition firings.
Simulation Loop for Cosmos/Simulink

Depending on the extracted event:

1. If it is a net transition firing:
   - Update marking;
   - Execute C code;
   - Generate new net events possibly including Simulink input transition firings.

2. If it is a Simulink input transition firing:
   - Update Simulink signals;
   - Set up the Simulink event to current time.
Simulation Loop for Cosmos/Simulink

Depending on the extracted event:

1. If it is a net transition firing:
   - Update marking;
   - Execute C code;
   - Generate new net events possibly including Simulink input transition firings.

2. If it is a Simulink input transition firing:
   - Update Simulink signals;
   - Set up the Simulink event to current time.

3. If it is the Simulink event:
   - Evaluate Simulink signals;
   - Put the Simulink event back at next Simulink simulation step;
   - Add Simulink output transition firings to event queue.
Simulation Loop for Cosmos/Simulink

Depending on the extracted event:

1. If it is a net transition firing:
   - Update marking;
   - Execute C code;
   - Generate new net events possibly including Simulink input transition firings.

2. If it is a Simulink input transition firing:
   - Update Simulink signals;
   - Set up the Simulink event to current time.

3. If it is the Simulink event:
   - Evaluate Simulink signals;
   - Put the Simulink event back at next Simulink simulation step;
   - Add Simulink output transition firings to event queue.

4. If it is a Simulink output transition firing:
   - Update output places of this transition.
Outline

1. Semantics for Simulink

2. Integrating Simulink into Cosmos

3. Benchmarks
A Model for Room Temperature

Single room with two heaters trying to keep temperature between 20°C and 25°C

Simulink model

- Differential equations that compute the current temperature
- External temperature: sine wave 5-25°C
- Hysteresis blocks triggered off when above 25°C

Stochastic net

- Heater failures triggered by stochastic transitions;
- A single repairman;
- Repairing through transitions with deterministic time.

One-way communication: state of heaters transmitted to the Simulink model.
Model

\[
\begin{align*}
F_1, F_2 : \mathbb{B}
\end{align*}
\]
Execution of the model

This data has been produced by COSMOS
Benchmarks

Performance Indices:
\( I_1 \): average min. temperature
\( I_2 \): elapsed time at target temperature
\( I_3 \): average temperature
\( I_4 \): average activity of heaters
\( I_5 \): average repairman idling rate

Linear Hybrid Automaton (for Properties):
\[
\begin{align*}
\dot{tc} &= 0 \\
\dot{curr} &= 1 \\
T &\geq 20 \\
\dot{tc} &= 1 \\
\dot{curr} &= 1 \\
T &< 20 \\
\text{All} &\implies \# \text{, } t = S_{time}
\end{align*}
\]

Indices

<table>
<thead>
<tr>
<th>Indices</th>
<th>Original</th>
<th>Discrete Integration</th>
</tr>
</thead>
<tbody>
<tr>
<td>( I_1 )</td>
<td>[ 18.69 ; 18.72 ]</td>
<td>[ 18.27 ; 18.29 ]</td>
</tr>
<tr>
<td>( I_2 )</td>
<td>[ 66.34 ; 67.22 ]</td>
<td>[ 92.92 ; 93.93 ]</td>
</tr>
<tr>
<td>( I_3 )</td>
<td>[ 22.43 ; 22.45 ]</td>
<td>[ 22.48 ; 22.49 ]</td>
</tr>
<tr>
<td>( I_4 )</td>
<td>[ 0.499 ; 0.501 ]</td>
<td>[ 0.488 ; 0.489 ]</td>
</tr>
<tr>
<td>( I_5 )</td>
<td>[ 0.923 ; 0.925 ]</td>
<td>[ 0.923 ; 0.925 ]</td>
</tr>
</tbody>
</table>

Models

<table>
<thead>
<tr>
<th>Models</th>
<th>Build time</th>
<th>Sim. time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Orig</td>
<td>5.74s</td>
<td>6 885s</td>
</tr>
<tr>
<td>DTI</td>
<td>5.73s</td>
<td>1 145s</td>
</tr>
<tr>
<td>noSlx</td>
<td>1.31s</td>
<td>2s</td>
</tr>
</tbody>
</table>
Conclusion and Future Work

What has been done

- Simultaneous simulation of a stochastic net and a Simulink model;
- Possible Simulink standalone execution.

Future developments

- Increasing the number of supported blocks;
- Wrappers for multi-model simulation;
- Adding floating-point color support for nets.

Planned applications

- Evaluation of autonomous vehicle controllers into various vehicular environments;
- Analysis of strategies for energy consumption in data centers.
Thank you for your attention!