Hybrid Cosimulation: It’s About Time

MODPROD 2016, Linköping

David Broman
KTH and UC Berkeley

Contributors of the presented work
• David Broman, KTH and UC Berkeley
• Christopher Brooks, UC Berkeley
• Fabio Cremona, UC Berkeley
• Lev Greenberg, IBM Research
• Marten Lohstroh, UC Berkeley
• Edward A. Lee, UC Berkeley
• Michael Masin, IBM Research
• Stavros Tripakis, UC Berkeley and Aalto
• Michael Wetter, Lawrence Berkeley N. Lab

Agenda

Part I
Functional Mockup Interface (FMI)

Part II
Hybrid Cosimulation with Superdense Integer Time
Part I
Functional Mock-up Interface (FMI)

What is FMI?

Functional Mock-up Interface (FMI) is a standard, not a tool.

- Used for simulating the dynamics of complex heterogeneous systems.
- Initiative from Daimler AG. Developed in a EU project called MODELISAR. Now maintained by Modelica Association.
- Current version is 2.0. Supported by more than 50 tools (open source and commercial).

Functional Mock-Up Unit (FMU) is a model instance that can be used in a simulation.

- An FMU is a zip file containing:
  - An XML file describing static info (e.g., port names)
  - C-files and dynamically loadable libraries implement the behavior.

The MA orchestrates the execution of the FMUs.

Master algorithms are not part of the standard. It is “up to the tool” to implement them.
This talk concerns FMI for co-simulation

### FMI 2.0 for co-simulation cannot model reactive systems

\[
\begin{align*}
\text{init}_c &: \mathbb{R} \geq 0 \to S_c \\
\text{set}_c &: S_c \times U_c \times \mathbb{V} \to S_c \\
\text{get}_c &: S_c \times Y_c \to \mathbb{V} \\
\text{doStep}_c &: S_c \times \mathbb{R} \geq 0 \to S_c \times \mathbb{R} \geq 0
\end{align*}
\]

- **Set** input values and **get** output values at distinct communication points.
- **doStep** advances the state and the time.

Version 2.0 makes it impossible to implement a component with zero latency (e.g., cannot implement synchronous components).

There is an initiative now to introduce a third kind of FMU: FMI for hybrid co-simulation.

Our work concerns a possible solution for such a standard.

---

"There is the additional restriction in "slaveInitialized" state that it is not allowed to call fmi2GetXXX functions after fmi2SetXXX functions without an fmi2DoStep call in between."

(FMI standard 2.0, July 25, 2014, page 104)

"... communication step size (hc). The latter must be > 0.0"  
(FMI standard 2.0, July 25, 2014, page 100)
Superdense Time

Non-negative real numbers represent time in usual Newtonian sense.

\[ T = \mathbb{R}_+ \times \mathbb{N} \]

Set of natural numbers

\[ \tau \in T \quad \tau = (t, n) \]

Every communication point is a member of \( T \) and \( T \) is totally ordered.

\( n \) is called the microstep, which indexes sequences of values at time \( t \).
Implementing Time

Specification uses real numbers, which is often approximated using floating-point numbers in implementations.

Problem: not safe to compare for equality (necessary in for instance a discrete-event formalism)

double r = 0.8;
double k = 0.7;
k = k + 0.1;
printf("%f,%f,%d\n", r, k, r==k);

A solution should provide the following properties:

1. The time origin should not affect the precision.
2. Addition of time must be associative.

These items do not hold for floating-point numbers.

r == k is false.

Continuous-Time (CT) and Discrete-event (DE) Signals

A signal $x$ is a function of time.

$$x: T \to \mathbb{R} \cup \{\varepsilon\}$$

Note: An implementation do NOT need to implement absent explicitly.

A continuous-time (CT) signal is one that has a non-absent value for all $\tau \in T$.

A discrete-event (DE) signal is one that has a non-absent value at only some $\tau \in D \subset T$, where $D$ is a discrete set.

A signal is discontinuous at any time $t \in \mathbb{R}$ if there exist $n, m \in \mathbb{N}$ such that $x(t, n) \neq x(t, m)$.
Test Case: Simultaneous Events

![Diagram of Simultaneous Events]

Test Case: Integrating Discontinuous Signals

![Diagram of Integrating Discontinuous Signals]
What should the time resolution be?

Who determines the resolution?
The MA? The FMUs? The user?

The FMU tells the MA what it prefers.

The MA tells the FMU what it should use.

Communication step size as an integer, instead of a floating-point number.

The MA states the resolution.

Negotiation. MA uses the highest resolution among all FMUs.

Also need to handle FMUs with floating-point time.

The MA must perform translations.

The FMU does not consider time (state update or output).
Conclusions

Some key take away points:

- The current FMI standard 2.0 lacks the possibility of hybrid co-simulation.
- A possible extension can be based on superdense time and integer time with negotiation of resolution between FMUs and MA.