Model verification and debugging of EOO models aided by model reduction techniques

Anton Sodja, Borut Zupančič

Univerza
v Ljubljani

Fakulteta
za elektrotehniko

EOOLT'10, Oslo, 3. 10. 2010

Introduction

Model verification assures that implemented model corresponds to the conceptual model.

Most model verification techniques originates in verification and debugging of computer programs – a traditional way of model implementation.

Many traditional verification/debugging approaches and tools can not be used with declarative EOO modeling languages, because of discrepancies between model representation and model's computational (simulation) form.

The lack of efficient verification/debugging tools hinders building and maintaining of large and complex models.

Verification techniques of EOO modeling

Using only model (static analysis):

- checking if model is well-constrained (numbers of equation and variables are equal)
- unit checking
- . . .

Using model and simulation results (dynamic analysis):

- comparing analytical calculations or measurements with simulation results
- checking that some model assumptions are not violated by actual conditions during simulation (assert statements)
- automated regression tests
- . . .

However, most modeling environments using Modelica does not provide any integrated tool for localisation of errors detected in simulation results.

Cognitive aspects of model verification

Models represent a knowledge about a system in a meaningful way and should facilitate learning and drawing conclusion about behavior of the system.

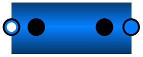
Models can be represented in two ways in EOO modeling languages which are close to the users' perception of the systems:

- system of (acausal) equations – derived from first principles
- graphical connection graph (set of related/interacting (sub)models) – reflecting system's topology

Specifics of EOO modeling approach are libraries of models, which can not be validated because they do not represent any particular real system.
On the other hand, verification is very important, because user of the library rely that component from the library corresponds to its documentation.

However, an implementation of general libraries may demand complicated inheritance and replaceable schemes.

Example: implementation complexity

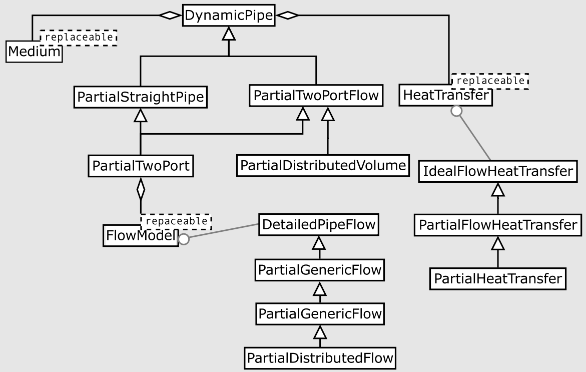


$$\frac{\partial(\rho\, A)}{\partial t}+\frac{\partial(\rho\, A\, v)}{\partial x}=0$$

$$\frac{\partial(\rho\, v\, A)}{\partial t}+\frac{\partial(\rho\, v^2\, A)}{\partial x}=-A\,\frac{\partial p}{\partial x}-\frac{1}{2}\,\rho\, v\,|v|\, f\, S-A\,\rho\, g\,\frac{\partial z}{\partial x}$$

$$\frac{\partial(\rho\,(u+\frac{v^2}{2})\, A)}{\partial t}+\frac{\partial(\rho\, v\,(u+\frac{p}{\rho}+\frac{v^2}{2})\, A)}{\partial x}=-A\,\rho\, v\, g\,\frac{\partial z}{\partial x}+\frac{\partial}{\partial x}(kA\,\frac{\partial T}{\partial x})+\dot{Q}$$

Inheritance scheme of Modelica.Fluid.Pipes.DynamicPipe



Model order reduction techniques

Engineers use their intuition and experience to determine the components that most affect the system performance.

This approach can be supported by model reduction techniques which generate proper models – models which retain physical meaningful components.

All (quantitative) model reduction techniques consist of three steps:

- choosing appropriate excitation signal and running a series of simulations
- ranking the individual coordinates or elements by the appropriate metrics
- removing those that fall below a certain threshold

Connection graphs can be reduced by techniques using energy-based metric.

System of equations can be reduced by various symbolic approximation techniques.

Energy-based metrics for model reduction

Basic idea

Submodels possessing low energy flows flowing into them have also low impact on system's dynamics (response).

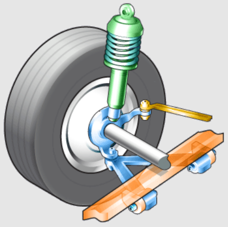
A modeling formalism based on power/energy is required – bond graphs, Lyapunov function.

Some used metrics:

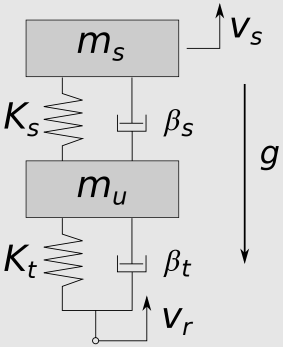
- rms power of bonds
- merits of activity – time integral of absolute value of power
- energy associated relative to neighboring bonds
- value of Lyapunov function for each state (coordinate)

Example: Car suspension

Car suspension



Scheme of the system

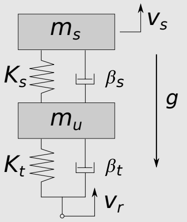


m_s	sprung mass
m_u	unsprung mass
K_s	suspension stiffness
K_t	tire stiffness
β_s	suspension damping
β_t	tire damping

v_s	vertical velocity of sprung mass
v_r	road/tire interface
g	gravity

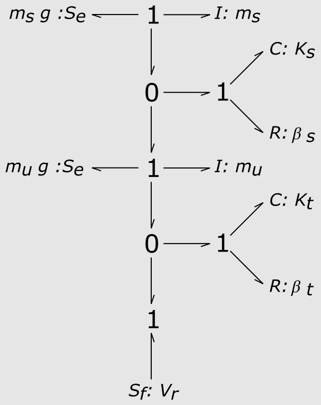
Energy-based reduction of bond graphs

Scheme of the system



Rank	Element	Index	Cumulative index
1	K_s	60.02%	60.02%
2	m_s	24.33%	84.36%
3	β_s	6.79%	91.46%
4	K_t	6.65%	97.80%
5	m_u	2.19%	99.98%
6	β_t	0.02%	100.00%

Corresponding bond graph

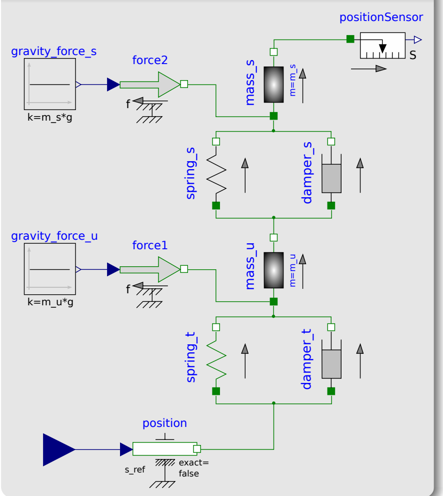


Similarities between bond-graph and Modelica models

Bond-graphs and Modelica graphical connection schemes are similar in some aspects:

- acausal description
- topology of the system is preserved
- connector consists of *effort* and *flow* variable
- energy flow can be associated with connections

Car-suspension model in Modelica



Modelica Standard Library connectors

Modelica Standard Library covers most domains of physical modeling.

Modelica.Electric

```
connector Pin
  SI.Voltage v;
  flow SI.Current i;
end HeatPort;
```

$$P = v \cdot i$$

Modelica.Magnetic

```
connector MagneticPort
  SI.MagneticPotentialDifference V_m;
  flow SI.MagneticFlux Phi;
end MagneticPort;
```

$$P = V_m \cdot \Phi$$

Modelica Standard Library connectors (cont.)

Modelica.Mechanics.Multibody

```
connector Frame
  SI.Position r_0[3];
  Frames.Orientation R;
  flow SI.Force f[3];
  flow SI.Torque t[3];
end Frame;
```

$$P = \frac{d}{dt}(\mathbf{T}\,\mathbf{r_o}) \cdot \mathbf{f} + \boldsymbol{\omega} \cdot \mathbf{t}$$

Modelica.Mechanics.Rotational

```
connector Flange_a
  SI.Angle phi;
  flow SI.Torque tau;
end Flange_a;
```

$$P = \frac{d}{dt}\phi \cdot \tau$$

Modelica.Mechanics.Translational

```
connector Flange_a
  SI.Position s;
  flow SI.Force f;
end Flange_a;
```

$$P = \frac{d}{dt}s \cdot f$$

Modelica Standard Library connectors (cont.)

Modelica.Fluid

```
connector FluidPort
  replaceable package Medium =
  Modelica.Media.Interfaces.PartialMedium;

  flow Medium.MassFlowRate m_flow;
  Medium.AbsolutePressure p;
  stream Medium.SpecificEnthalpy
h_outflow;
  stream Medium.MassFraction
Xi_outflow[Medium.nXi];
end FluidPort;
```

$$P = \dot{m} \cdot h + p \cdot \dot{m}/\rho + \sum \mu_i \cdot \dot{N}_i$$

Modelica.Thermal.HeatTransfer

```
connector HeatPort
  SI.Temperature T;
  flow SI.HeatFlowRate Q_flow;
end HeatPort;
```

$$P = \dot{Q}_{flow}$$

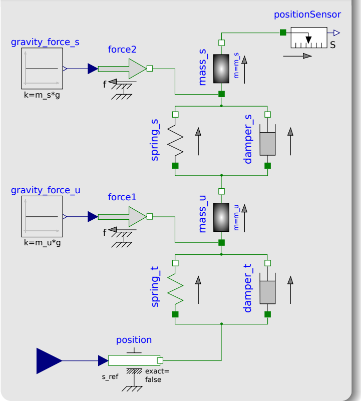
Adaptation of bond-graphs-based energy metric to Modelica models

- a connection can be assigned with energy-flow — seen as a quasi-bond:
  $\rightsquigarrow$
- Connection nodes can be treated as 0-junctions: sum of flow variables is zero
- to each component instance of the scheme, a $\tilde{1}$-junction can be assigned:
  - sum of energy flow from all connectors and energy flow into the component equals zero (energy conservation law)
  - in $\tilde{1}$-junction sum of effort variables is not (necessarly) zero!

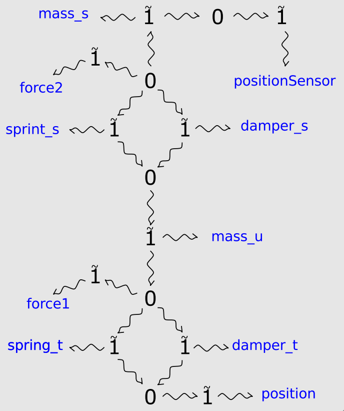
Quasi-bonds

A quasi-bond graph can be constructed from Modelica connection graphs:

Block scheme in Modelica



Quasi-bond graph



Approximation of symbolic equations

Each (sub)model in Modelica can be flattened to a set of DAE equations. Simulation results can be used to rank importance of equations' terms with respect to the variable(s) of interest.

With a bit more complicated (sub)models, symbolic approximation techniques are needed to remain listing of flattened model useful.

A ranking metric can be defined as numerical error with respect to the variables of interest.
Example of some simplification/approximation techniques:

Example: algebraic simplification

$$\mathbf{x_1} + 2\,x_2 = x_3$$

$$x_2 - x_3 = a$$

can be replaced by

$$x_1 + x_3 + 2\,a = 0$$

Example: term deletion

$$\mathbf{x_1} = a_1 \cdot x_2 + a_2 \cdot x_3$$

might be replaced by

$$x_1 = a_1 \cdot x_2$$

Also linearization might be used in some cases.

Summary

There is a lack of integrated verification/debugging tools in EOO modeling environments.

Ranking methods normally used in model reduction techniques could be used to help user focus on components which significantly contribute to specific simulation variables' trajectories.

Ranking of model components (submodels, terms of equations) is close to human reasoning when dealing with complex systems.

This approach can also help answer to some other questions, e.g., *"Is my model too complicated?"*

However, energy-based metrics are dependent on physical properties of the system and thus their use is limited.

The current approach does not treat hybrid systems adequately.