Modelica Model Debugging

Martin Sjölund <martin.sjolund@liu.se>
Linköping University, Sweden

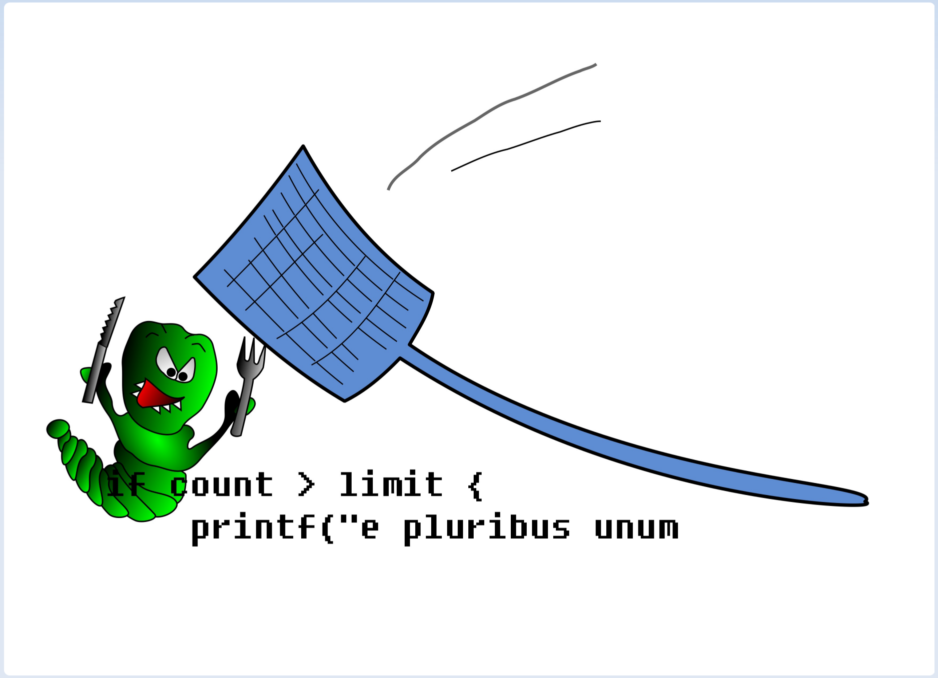
OpenModelica Workshop
February 2013, Linköping, Sweden

Modelica

- No explicit control flow
- Optimization
- Symbolic manipulations
- Numerical methods and solvers
- Linear/Non-linear blocks
- Events

Modelica Debugging

- Need knowledge
  - Modelica
  - The tool
  - Numerical methods



Typical Error Message

```
Error solving nonlinear system 132
    time = 0.002
    residual[0] = 0.288956
    x[0] = 1.105149
    residual[1] = 17.000400
    x[1] = 1.248448
    ...
```

Better Error Message

```
Error solving nonlinear system 132 <more info>
    time = 0.002
    residual[0] = 0.288956
    x[0] = 1.105149
    residual[1] = 17.000400
    x[1] = 1.248448
    ...
```

Origin

- Several Levels
  - (Graphical Representation)
  - Source Code
  - Flat Equation-System
  - Optimized Equation-System
  - Translated Code (typically C)
- It should always be possible to go backwards
  - Simple for flattened equation system to source
  - Harder for optimized code

Symbolic Transformations

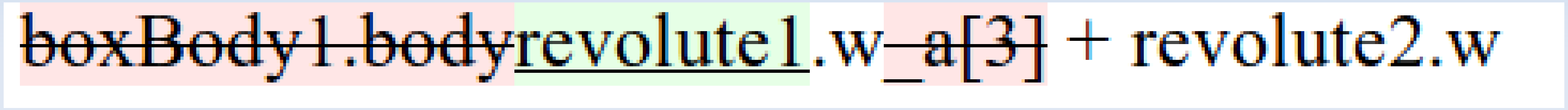
- From source code to flat equations
  - Most of the structure remains
  - Few symbolic manipulations (mostly simplification/evaluation)
- Equation System Optimization
  - Changes structure
  - Strong connected components
  - Variable replacements
  - … and more

Tracing Transformations

- Simple Idea
  - Store transformations as equation metadata
  - Works best for operations on single equations
- Each kind of transformation is different
  - Alias Elimination ($a = b$)
  - Gaussian Elimination (linear systems, several equations)
  - Equation solving ($f_1(a,b) = f_2(a,b)$, solve for a)
  - ...

Alias Elimination

- boxBody1.body.w_a[3] = revolute1.w
- Can remove one variable and replace it with the other


~~boxBody1.body~~revolute1.w~~_a[3]~~ + revolute2.w

Operations

- Simplify
- Substitution
  - Alias elimination
  - Known variables
- Inline
- Scalarization
- Differentiation
- Solve w.r.t.
- Solve linear system symbolically
- New dummy derivative added
- Residual form

Debugging Using the Trace

- General Purpose
  - Verify performance and correctness of the trace
  - Navigate equations
    - Cross-referencing
    - Go to parents
    - View trajectories
- Special-Purpose
  - Non-linear system debugger

Trace Example

Demo

+simCodeTarget=Dump

Future Work

- Graphical debugger
  - General-purpose
  - Domain-specific
- Cross-references, parent blocks
- Runtime support to launch debugger
- Tracing in algorithmic code
- More operations recorded
  - Control flow and events
  - Forgotten optimization modules