ThermoPower Library Simulation

Francesco Casella
(francesco.casella@polimi.it)

Dipartimento di Elettronica, Informazione e Bioingegneria
Politecnico di Milano
Introduction

• Back in 2002, work started at Politecnico di Milano on the modelling of power plants using the Modelica language
• The tool we used back then was Dymola 5.3, using Modelica 2.0
• Modelica_Media was still under development, so the library used Hubertus Tummescheit's ThermoFluid library for water/steam properties
Introduction

• Back in 2002, work started at Politecnico di Milano on the modelling of power plants using the Modelica language
• The tool we used back then was Dymola 5.3, using Modelica 2.0
• Modelica_Media was still under development, so the library used Hubertus Tummescheit's ThermoFluid library for water/steam properties
• Back in 2002, work started at Politecnico di Milano on the modelling of power plants using the Modelica language
• The tool we used back then was Dymola 5.3, using Modelica 2.0
• Modelica_Media was still under development, so the library used Hubertus Tummescheit's ThermoFluid library for water/steam properties
• The library evolved together with the Modelica language
  – Modelica.Media
  – Modelica 3.x
  – Stream connectors
  – Homotopy-based initialization
• In the meantime, OpenModelica has evolved
  – from an exotic Computer Science project with lots of limitations
  – to a full-fledged, fully Modelica 3.2r2 compliant tool
• We never thought of investing time to develop a **downgraded** version of ThermoPower to work with OpenModelica
• The tool needs to catch up, not the library!
Introduction

• Back in 2002, work started at Politecnico di Milano on the modelling of power plants using the Modelica language
• The tool we used back then was Dymola 5.3, using Modelica 2.0
• Modelica_Media was still under development, so the library used Hubertus Tummescheit's ThermoFluid library for water/steam properties
• The library evolved together with the Modelica language
  – Modelica.Media
  – Modelica 3.0
  – Stream connectors
  – Homotopy-based initialization
• In the meantime, OpenModelica has evolved
  – from an exotic Computer Science project with lots of limitations
  – to a full-fledged, fully Modelica 3.2r2 compliant tool
• We never thought of investing time to develop a *downgraded* version of ThermoPower to work with OpenModelica
• The tool needs to catch up, not the library!

Has the moment finally arrived?
The Good News
ThermoPower is handled by OMEdit
ThermoPower is handled by OMEdit (with some glitches...)

![ThermoPower model in OMEdit](image)
Models can be simulated, producing the right results
Models can be simulated, producing the right results
Non-trivial models are simulated correctly

- CISE plant model (see Modelica Conference 2003 paper)
Non-trivial models are simulated correctly

- Model with about 40 states and 500 non-trivial algebraic variables compiled and simulated correctly
- Large initialization problem with 132 coupled nonlinear equations solved successfully
The majority of test cases from the test suite now work.

Recursive BuildModel Test

BuildModel Results: 70/97 succeeded
Simulation Results: 52/97 succeeded
Total time taken: 17m58s
OpenModelica Version: 1.9.1-dev (r18935) (RML version)
Test started: 2014-02-02 08:37:33
Tested Library: ThermoPower 3.1 revision 234
BuildModel time limit: 900s
Simulation time limit: 480s
Default tolerance: 1e-06
Flags:
setCommandLineOptions("+d=nogen,initialization");
setMatchingAlgorithm("PFPlusExt");
Summary of good news

- The ThermoPower library can be handled by OMEdit
  - with some glitches that will be discussed next
- Many models can be simulated, obtaining correct results
  - simple test cases
  - non-trivial examples
- Steady-state initialization works well even for non-trivial examples
  - new initialization solver with symbolic processing in place
  - homotopy transformations supported
  - *automatic elimination of redundant & consistent initial equations soon available*
- *Interactive debugger for equation-based models almost ready for use*
  - *still missing support for thorough debugging of solver failures for nonlinear equations solved by tearing*
  - *still missing support for initialization solver failures during homotopy transformations*
The Not-So-Good News
CPU time to compute Modelica.Media medium properties

- Results obtained by algebraic models, 10000 steps with Euler's algorithm
- Only pure simulation time reported
- Checked proportionality with number of steps (no one-time overhead)

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Sim. Time OMC</th>
<th>Sim. Time Dymola</th>
</tr>
</thead>
<tbody>
<tr>
<td>WaterIF97Efficiency.Test1</td>
<td>0.85</td>
<td>0.15</td>
</tr>
<tr>
<td>WaterIF97Efficiency.Test2</td>
<td>2.05</td>
<td>0.15</td>
</tr>
<tr>
<td>WaterIF97Efficiency.Test3</td>
<td>0.82</td>
<td>0.15</td>
</tr>
<tr>
<td>WaterIF97Efficiency.Test4</td>
<td>5.46</td>
<td>1.47</td>
</tr>
<tr>
<td>WaterIF97Efficiency.Test5</td>
<td>0.42</td>
<td>0.15</td>
</tr>
<tr>
<td>WaterIF97Efficiency.Test6</td>
<td>4.25</td>
<td>1.41</td>
</tr>
<tr>
<td>IdealGasEfficiency.Test1</td>
<td>0.08</td>
<td>0.12</td>
</tr>
<tr>
<td>IdealGasEfficiency.Test2</td>
<td>0.06</td>
<td>0.13</td>
</tr>
</tbody>
</table>
WaterIF97Efficiency.Test1

model Test1 "Compute density of water via state record"
Water.ThermodynamicState state;
Water.AbsolutePressure p;
Water.Density rho;
Water.SpecificEnthalpy h;
equation
  p = 1e5;
  h = 1e5 + time*1e5;
  state = Water.setState_ph(p,h);
  rho = Water.density(state);
end Test1;

- IF97 properties computed via setState_ph() function
- most of the CPU time spent solving the equations of state when calling setState_ph()
- only density retrieved
- OMC almost 6X slower than Dymola

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Sim. Time OMC</th>
<th>Sim. Time Dymola</th>
</tr>
</thead>
<tbody>
<tr>
<td>WaterIF97Efficiency.Test1</td>
<td>0.85</td>
<td>0.15</td>
</tr>
</tbody>
</table>
**WaterIF97Efficiency.Test2**

```plaintext
model Test2 "Compute multiple properties of water via state record"
    extends Test1;
    Water.Temperature T;
    Water.SpecificEntropy s;
    Water.DerDensityByPressure ddph;
    Water.DerDensityByEnthalpy ddhp;

equation
    T = Water.temperature(state);
    s = Water.s specificEntropy(state);
    ddph = Water.density_derp_h(state);
    ddhp = Water.density_derh_p(state);
end Test2;
```

- IF97 properties computed via setState_ph() function
- four more properties retrieved
- most of the CPU time should spent solving the equations of state when calling setState_ph()
- OMC almost 3X slower than previous case → unwanted repeated comp.
- probably something goes wrong with Inline/LateInline

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Sim. Time OMC</th>
<th>Sim. Time Dymola</th>
</tr>
</thead>
<tbody>
<tr>
<td>WaterIF97Efficiency.Test1</td>
<td>0.85</td>
<td>0.15</td>
</tr>
<tr>
<td>WaterIF97Efficiency.Test2</td>
<td>2.05</td>
<td>0.15</td>
</tr>
</tbody>
</table>
**WaterIF97Efficiency.Test3**

```model
model Test3 "Compute density of water via direct function call"
Water.AbsolutePressure p;
Water.Density rho;
Water.SpecificEnthalpy h;
equation
  p = 1e5;
  h = 1e5 + time*1e5;
  rho = Water.density(Water.setState_ph(p,h));
end Test3;
```

- similar to Test1, no significant differences reported

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Sim. Time OMC</th>
<th>Sim. Time Dymola</th>
</tr>
</thead>
<tbody>
<tr>
<td>WaterIF97Efficiency.Test1</td>
<td>0.85</td>
<td>0.15</td>
</tr>
<tr>
<td>WaterIF97Efficiency.Test3</td>
<td>0.82</td>
<td>0.15</td>
</tr>
</tbody>
</table>
WaterIF97Efficiency.Test4

```model Test4
  "Compute multiple properties of water via direct function call"
  extends Test3;
  Water.Temperature T;
  Water.SpecificEntropy s;
  Water.DerDensityByPressure ddph;
  Water.DerDensityByEnthalpy ddhp;
  equation
    T = Water.temperature(Water.setState_ph(p,h));
    s = Water.specificEntropy(Water.setState_ph(p,h));
    ddph = Water.density_derivative_h(Water.setState_ph(p,h));
    ddhp = Water.density_derivative_p(Water.setState_ph(p,h));
  end Test4;
```

- if Common Subexpression Elimination was applied correctly, same CPU time as Test 1
- Apparently CSE is applied by neither tool

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Sim. Time OMC</th>
<th>Sim. Time Dymola</th>
</tr>
</thead>
<tbody>
<tr>
<td>WaterIF97Efficiency.Test3</td>
<td>0.82</td>
<td>0.15</td>
</tr>
<tr>
<td><strong>WaterIF97Efficiency.Test4</strong></td>
<td><strong>5.46</strong></td>
<td><strong>1.47</strong></td>
</tr>
</tbody>
</table>
WaterIF97Efficiency.Test5

- Test 5 should be perfectly equivalent to Test1
- For some reason, it works 2X fast as Test1 in OMC
- Still 3X slower than Dymola

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Sim. Time OMC</th>
<th>Sim. Time Dymola</th>
</tr>
</thead>
<tbody>
<tr>
<td>WaterIF97Efficiency.Test1</td>
<td>0.85</td>
<td>0.15</td>
</tr>
<tr>
<td><strong>WaterIF97Efficiency.Test5</strong></td>
<td><strong>0.42</strong></td>
<td><strong>0.15</strong></td>
</tr>
</tbody>
</table>

```model Test5 "Compute density of water via direct function call"
  Water.AbsolutePressure p;
  Water.Density rho;
  Water.SpecificEnthalpy h;
equation
  p = 1e5;
  h = 1e5 + time*1e5;
  rho = Water.density_ph(p,h);
end Test5;```
WaterIF97Efficiency.Test6

model Test6
  "Compute multiple properties of water via direct function call"
  extends Test3;
  Water.Temperature T;
  Water.SpecificEntropy s;
  Water.DerDensityByPressure ddph;
  Water.DerDensityByEnthalpy ddhp;
  equation
    T = Water.temperature_ph(p,h);
    s = Water.specificEntropy(Water.setState_ph(p,h));
    ddph = Water.density_derp_h(Water.setState_ph(p,h));
    ddhp = Water.density_derrh_p(Water.setState_ph(p,h));
end Test6;

- Slight variation of Test4
- Surprisingly somewhat faster in OMC
- Same considerations apply as for Test4

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Sim. Time OMC</th>
<th>Sim. Time Dymola</th>
</tr>
</thead>
<tbody>
<tr>
<td>WaterIF97Efficiency.Test1</td>
<td>0.85</td>
<td>0.15</td>
</tr>
<tr>
<td>WaterIF97Efficiency.Test4</td>
<td>5.46</td>
<td>1.47</td>
</tr>
<tr>
<td>WaterIF97Efficiency.Test6</td>
<td>4.25</td>
<td>1.41</td>
</tr>
</tbody>
</table>
IdealGasEfficiency.Test1 & Test2

- Ideal gas property computation
- `setState_pTX()` and equation-based `BaseProperties` tested
- OMC is 33% faster than Dymola (different accounting of overheads could be the cause)
- In steam power plant models, the bottleneck are the IF97 water/steam computations, not the ideal gas computations

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Sim. Time OMC</th>
<th>Sim. Time Dymola</th>
</tr>
</thead>
<tbody>
<tr>
<td>IdealGasEfficiency.Test1</td>
<td>0.08</td>
<td>0.12</td>
</tr>
<tr>
<td>IdealGasEfficiency.Test2</td>
<td>0.06</td>
<td>0.13</td>
</tr>
</tbody>
</table>
## CPU Time to Simulate Models

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Sim. Time OMC</th>
<th>Sim. Time Dymola</th>
</tr>
</thead>
<tbody>
<tr>
<td>TestFlow1DFV_A</td>
<td>7.4</td>
<td>0.42</td>
</tr>
<tr>
<td>CISESim120501</td>
<td>74.9</td>
<td>2.4</td>
</tr>
</tbody>
</table>
CPU-time for integration      : 0.424 seconds
CPU-time for one GRID interval: 0.848 milli-seconds
Number of result points       : 507
Number of GRID points         : 501
Number of (successful) steps  : 332
Number of F-evaluations       : 1962
Number of H-evaluations       : 835
Number of Jacobian-evaluations: 66
Number of (model) time events : 3
Number of (U) time events     : 0
Number of state events        : 0
Number of step events         : 0
Minimum integration stepsize  : 1.21e-007
Maximum integration stepsize  : 2.88
Maximum integration order     : 5

LOG_STATS | info    | timer
|----------|---------|-------------------------------------
|          |         | 0.00163591s [ 0.0%] pre-initialization
|          |         | 0.112609s [ 1.5%] initialization
|          |         | 0.00412725s [ 0.1%] steps
|          |         | 0.0271429s [ 0.4%] creating output-file
|          |         | 0.00924873s [ 0.1%] event-handling
|          |         | 0.00700192s [ 0.1%] overhead
|          |         | 7.43596s [ 97.9%] simulation
|          |         | 7.59773s [100.0%] total

LOG_STATS | info    | events
|----------|---------|-----------------
|          |         | 3 state events
|          |         | 0 time events

LOG_STATS | info    | solver: DASSL
|----------|---------|-----------------
|          |         | 344 steps taken
|          |         | 460 calls of functionODE
|          |         | 78 evaluations of jacobian
|          |         | 3 error test failures
|          |         | 0 convergence test failures
TestFlow1DFV_A – DASSL – tol = 1e–6

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU-time for integration</td>
<td><strong>0.424 seconds</strong></td>
</tr>
<tr>
<td>CPU-time for one GRID interval</td>
<td>0.848 milli-seconds</td>
</tr>
<tr>
<td>Number of result points</td>
<td>507</td>
</tr>
<tr>
<td>Number of GRID points</td>
<td>501</td>
</tr>
<tr>
<td>Number of (successful) steps</td>
<td>332</td>
</tr>
<tr>
<td>Number of F-evaluations</td>
<td>1962</td>
</tr>
<tr>
<td>Number of H-evaluations</td>
<td>835</td>
</tr>
<tr>
<td>Number of Jacobian-evaluations</td>
<td>66</td>
</tr>
<tr>
<td>Number of (model) time events</td>
<td>3</td>
</tr>
<tr>
<td>Number of (U) time events</td>
<td>0</td>
</tr>
<tr>
<td>Number of state events</td>
<td>0</td>
</tr>
<tr>
<td>Number of step events</td>
<td>0</td>
</tr>
<tr>
<td>Minimum integration stepsize</td>
<td>1.21e-007</td>
</tr>
<tr>
<td>Maximum integration stepsize</td>
<td>2.88</td>
</tr>
<tr>
<td>Maximum integration order</td>
<td>5</td>
</tr>
</tbody>
</table>

**Dymola**

**OMC**

<table>
<thead>
<tr>
<th>Log Statistics</th>
<th>Events</th>
<th>Solver: DASSL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Information</td>
<td></td>
<td>Steps taken</td>
</tr>
<tr>
<td>Timer</td>
<td></td>
<td>344 steps</td>
</tr>
<tr>
<td></td>
<td></td>
<td>460 calls ODE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>78 evaluations</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 error failures</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 convergence failures</td>
</tr>
</tbody>
</table>
TestFlow1DFV_A – DASSL – tol = 1e–6

CPU-time for integration : 0.424 seconds
CPU-time for one GRID interval: 0.848 milli-seconds
Number of result points : 507
Number of GRID points : 501
Number of (successful) steps : 332
Number of F-evaluations : 1962
Number of H-evaluations : 835
Number of Jacobian-evaluations: 66
Number of (model) time events : 3
Number of (U) time events : 0
Number of state events : 0
Number of step events : 0
Minimum integration stepsize : 1.21e-007
Maximum integration stepsize : 2.88
Maximum integration order : 5

LOG_STATS | info | timer
|          |      | 0.00163591s [ 0.0%] pre-initialization
|          |      | 0.112609s [ 1.5%] initialization
|          |      | 0.00412725s [ 0.1%] steps
|          |      | 0.0271429s [ 0.4%] creating output-file
|          |      | 0.00924873s [ 0.1%] event-handling
|          |      | 0.00700192s [ 0.1%] overhead
|          |      | 7.43596s [97.9%] simulation
|          |      | 7.59773s [100.0%] total

LOG_STATS | info | events
|          |      | 3 state events
|          |      | 0 time events

LOG_STATS | info | solver: DASSL
|          |      | 344 steps taken
|          |      | 168 calls of functionODE
|          |      | 78 evaluations of jacobian
|          |      | 3 error test failures
|          |      | 0 convergence test failures
TestFlow1DFV_A – DASSL – tol = 1e–6

CPU-time for integration : 0.424 seconds
CPU-time for one GRID interval: 0.848 milli-seconds
Number of result points : 507
Number of GRID points : 501
Number of (successful) steps : 332
Number of F-evaluations : 1962
Number of H-evaluations : 835
Number of Jacobian-evaluations: 66
Number of (model) time events : 3
Number of state events : 0
Number of step events : 0
Minimum integration stepsize : 1.21e-007
Maximum integration stepsize : 2.88
Maximum integration order : 5

LOG_STATS | info          | timer
-----------|---------------|--------------------
            |               | 0.00163591s [ 0.0%] pre-initialization
            |               | 0.112609s [ 1.5%] initialization
            |               | 0.00412725s [ 0.1%] steps
            |               | 0.0271429s [ 0.4%] creating output-file
            |               | 0.00924873s [ 0.1%] event-handling
            |               | 0.00700192s [ 0.1%] overhead
            |               | 7.43596s [ 97.9%] simulation
            |               | 7.59773s [100.0%] total

LOG_STATS | info | events
-----------|------|--------
            |      | 3 state events
            |      | 0 time events

LOG_STATS | info | solver: DASSL
-----------|------|------------------
            |      | 344 steps taken
            |      | 469 calls of functionODE
            |      | 78 evaluations of jacobian
            |      | 0 error test failures
            |      | 0 convergence test failures
CPU-time for integration : 2.4 seconds
CPU-time for one GRID interval: 2.4 milli-seconds
Number of result points : 1013
Number of GRID points : 1001
Number of (successful) steps : 243
Number of F-evaluations : 2294
Number of H-evaluations : 1249
Number of Jacobian-evaluations: 94
Number of (model) time events : 6
Number of (U) time events : 0
Number of state events : 0
Number of step events : 0
Minimum integration stepsize  : 0.0002
Maximum integration stepsize  : 52.4
Maximum integration order : 5

<table>
<thead>
<tr>
<th>LOG_STATS</th>
<th>info</th>
<th>timer</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0.00213257s [ 0.0%] pre-initialization</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.40867s [ 1.8%] initialization</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.0254833s [ 0.0%] steps</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.111198s [ 0.1%] creating output-file</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.0591024s [ 0.1%] event-handling</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.0296431s [ 0.0%] overhead</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>74.946s [ 97.9%] simulation</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>76.5823s [100.0%] total</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>LOG_STATS</th>
<th>info</th>
<th>events</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>6 state events</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 time events</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>LOG_STATS</th>
<th>info</th>
<th>solver: DASSL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>255 steps taken</td>
</tr>
<tr>
<td></td>
<td></td>
<td>316 calls of functionODE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>88 evaluations of jacobian</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 error test failures</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 convergence test failures</td>
</tr>
</tbody>
</table>
CPU-time for integration : 2.4 seconds
CPU-time for one GRID interval: 2.4 milli-seconds
Number of result points     : 1013
Number of GRID points       : 1001
Number of (successful) steps: 243
Number of F-evaluations     : 2294
Number of H-evaluations     : 1249
Number of Jacobian-evaluations: 94
Number of (model) time events: 6
Number of (U) time events   : 0
Number of state events      : 0
Number of step events       : 0
Minimum integration stepsize: 0.0002
Maximum integration stepsize: 52.4
Maximum integration order   : 5

LOG_STATS | info          | ### STATISTICS ###
LOG_STATS | info          | timer
|          |   | | | 0.00213257s [ 0.0%] pre-initialization
|          |   | | | 1.40867s [ 1.8%] initialization
|          |   | | | 0.0254833s [ 0.0%] steps
|          |   | | | 0.111198s [ 0.1%] creating output-file
|          |   | | | 0.0591024s [ 0.1%] event-handling
|          |   | | | 0.0296431s [ 0.0%] overhead
|          |   | | | 74.946s [ 97.9%] simulation
|          |   | | | 76.5823s [100.0%] total
LOG_STATS | info          | events
|          |   | | | 6 state events
|          |   | | | 0 time events
LOG_STATS | info          | solver: DASSL
|          |   | | | 255 steps taken
|          |   | | | 318 calls of functionODE
|          |   | | | 88 evaluations of jacobian
|          |   | | | 1 error test failures
|          |   | | | 0 convergence test failures
CPU-time for integration : 2.4 seconds
CPU-time for one GRID interval: 2.4 milli-seconds
Number of result points : 1013
Number of GRID points : 1001
Number of (successful) steps : 243
Number of F-evaluations : 2294
Number of H-evaluations : 1249
Number of Jacobian-evaluations: 94
Number of (model) time events : 6
Number of state events : 0
Number of (U) time events : 0
Number of step events : 0
Minimum integration stepsize  : 0.0002
Maximum integration stepsize  : 52.4
Maximum integration order : 5

LOG_STATS | info | ### STATISTICS ###
LOG_STATS | info | timer
| | | 0.00213257s [ 0.0%] pre-initialization
| | | 1.40867s [ 1.8%] initialization
| | | 0.0254833s [ 0.0%] steps
| | | 0.111198s [ 0.1%] creating output-file
| | | 0.0591024s [ 0.1%] event-handling
| | | 0.0296431s [ 0.0%] overhead
| | | 74.946s [ 97.9%] simulation
| | | 76.5823s [100.0%] total
LOG_STATS | info | events
| | | 6 state events
| | | 0 time events
LOG_STATS | info | solver: DASSL
| | | 255 steps taken
| | | 316 calls of functionODE
| | | 88 evaluations of jacobian
| | | 1 error test failures
| | | 0 convergence test failures
CPU-time for integration : 2.4 seconds
CPU-time for one GRID interval: 2.4 milli-seconds
Number of result points : 1013
Number of GRID points : 1001
Number of (successful) steps : 243
Number of F-evaluations : 2294
Number of H-evaluations : 1249
Number of Jacobian-evaluations : 94
Number of (model) time events : 6
Number of (U) time events : 0
Number of state events : 0
Number of step events : 0
Minimum integration stepsize : 0.0002
Maximum integration stepsize : 52.4
Maximum integration order : 5

LOG_STATS | info | ### STATISTICS ###
LOG_STATS | info | timer
| | | 0.00213257s [0.0%] pre-initialization
| | | 1.40867s [1.8%] initialization
| | | 0.0254833s [0.0%] steps
| | | 0.111198s [0.1%] creating output-file
| | | 0.0591024s [0.1%] event-handling
| | | 0.0296431s [0.0%] overhead
| | | 74.946s [97.9%] simulation
| | | 76.5823s [100.0%] total

LOG_STATS | info | events
| | | 6 state events
| | | 0 time events

LOG_STATS | info | solver: DASSL
| | | 255 steps taken
| | | 226 calls of functionODE
| | | 88 evaluations of jacobian
| | | 1 error test failures
| | | 0 convergence test failures
Issues with code generation

• Some models still generate errors in the back-end stages:
  – Error: Internal error Transformation Module PFPlusExt index Reduction Method Pantelides failed!
  – Errors in the compilation of the C-code, due to incorrect C-code being generated

• Sometimes problem with models that work in Dymola and fail in OMC are due to subtle issues, e.g.:
  – semantics of parameter Real p(fixed = false) = 0
    • earlier versions of Dymola ignored the binding, but OMC did not
    • in this case Dymola was not complying to the specification
    • library and Dymola have been fixed a few months ago
  – semantics of parameter Real p(start = 2);
    • in this case, Modelica 3.3 suggests to add a binding equation p=2 and issue a warning
    • OMC didn't do that → underdetermined initialization system
    • error discovered and fixed on Feb 1st, 2014
  – other, still undiscovered problems...
Issues with OMEdit
Main issues with the OMEdit GUI

• OMEdit applies extensive reformatting to the code
  – whitespace and carriage returns
  – numerical literals (e.g. 1e6 → 1000000)

Not possible to co-develop with OMC & Dymola
Main issues with the OMEdit GUI

• OMEdit applies extensive reformatting to the code
  – whitespace and carriage returns
  – numerical literals (e.g. 1e6 → 1000000)

Not possible to co-develop with OMC & Dymola

• Replaceable models are not handled by OMC
  – they are used for replaceable heat transfer models
  – they are used in many other places in the MSL

Not possible to build heat exchanger models using OMEdit's GUI
Main issues with the OMEdit GUI

• OMEdit applies extensive reformatting to the code
  – whitespace and carriage returns
  – numerical literals (e.g. $1e6 \rightarrow 1000000$)

Not possible to co-develop with OMC & Dymola

• Replaceable models are not handled by OMC
  – they are used for replaceable heat transfer models
  – they are used in many other places in the MSL

Not possible to build heat exchanger models using OMEdit's GUI

• It is not possible to apply or edit modifiers in hierarchically structured models using OMEdit's GUI
  – modifiers can only be applied in the textual mode
  – this requires to know the names of modified sub-components and parameters, which is inconvenient

Not possible to handle complex, structured models
Replaceable models
Replaceable models
Editing parameters in hierarchically structured models

- ThermoPower.PowerPlants.Simulators.SteamPlant_Sim1
Editing parameters in hierarchically structured models

• ThermoPower.PowerPlants.Simulators.SteamPlant_Sim1
Editing parameters in hierarchically structured models

- ThermoPower.PowerPlants.Simulators.SteamPlant_Sim1
Editing parameters in hierarchically structured models

- ThermoPower.PowerPlants.Simulators.SteamPlant_Sim1
Editing parameters in hierarchically structured models

- ThermoPower.PowerPlants.Simulators.SteamPlant_Sim1
Editing parameters in hierarchically structured models

• This corresponds to applying the following modifier in the textual view

\[
\text{HRSG.Examples.HRSG}\_3LRh \ hRSG( \\
\quad \text{HeatExchangersGroup(Ec2\_HP(fluidFlow(Nt=14))))}
\]

• Possibly mixed up with other pre-existing modifiers

• This feature doesn't need to be implemented in the same way as Dymola,

• But it is **essential** that OMEdit has way to edit the parameters of hierarchically structured models by using the GUI
Exploiting parallelism

• The CISESim120501 model is compiled to executable simulation code
  – by Dymola/Visual Studio in 7 seconds
  – by OMEdit/OMC/gcc in 30 seconds

• It is probably possible to reduce the compilation time gap (currently 4X) by exploiting parallel compilation of the various C source files (is it already done by default?)

• OMEdit should automatically retrieve from the OS the number of parallel threads it can run, and use all of them by default

• Earlier work by Sjolund and Casella regarding the parallelization of the solution of the DAEs into ODEs should also become part of the mainstream implementation of OMEdit

• Also the parallel computation of Jacobians should be investigated, and become part of the mainstream implementation of OMEdit
Conclusions

• After 10 years from the start of the development of ThermoPower, the (open source!) library can now be used in the (open source) OpenModelica environment

• Many models can already be simulated, and produce the correct results

• Steady-state initialization is working nicely on fairly complex models

• Compilation time is still about 4X slower than Dymola

• Simulation time for models involving water/steam properties is still about 10X slower than Dymola

• The OMEdit GUI should be improved regarding replaceable models and parameter editing in hierarchically structured models

• Parallel thread execution should be used more aggressively

• The declarative debugger (once polished up) will be a boon

• We might expect OMC to be a viable alternative to Dymola for the simulation of ThermoPower-based models by the end of 2014!
Thank you for your kind attention!