Using SPARK as a Solver for Modelica

Michael Wetter
Philip Haves
Michael A. Moshier
Edward F. Sowell

July 30, 2008
Overview

- Overview of SPARK, Modelica, OpenModelica, Dymola
- Problem reduction
- SPARK integration in OpenModelica
- SPARK/Dymola benchmark
- Future work
SPARK, Modelica, OpenModelica, Dymola
Modelica

- Modeling language developed since 1996 by Modelica Association
  - Model development environments
  - Solvers
- Designed by developers of
  - Allan, Dymola, NMF, ObjectMath, Omola, SIDOPS+, Smile
- Well positioned to become de-facto standard for modeling multi-engineering systems
  - e.g.: ITEA2 (Eurosystlib: 110 man years, 20 partners to develop Modelica libraries, but not for building HVAC systems).
- Supports differential, algebraic and discrete equations
- Declarative instead of imperative
- Object-oriented
- Automatic documentation
SPARK

- Developed since 1989 by LBNL and Ayres Sowell Associates
- Supports differential and algebraic equations
- Declarative instead of imperative
- Uses graph-theoretic methods to:
  - define well-posed problems
  - generate computationally efficient solution sequences
### Questions:

1) How can SPARK be used as a solver for OpenModelica?

2) How does SPARK compare with best in class Modelica simulator?
# Dymola/SPARK Comparison

## Features and availability

<table>
<thead>
<tr>
<th>Feature</th>
<th>Dymola</th>
<th>SPARK</th>
</tr>
</thead>
<tbody>
<tr>
<td>Partitioning</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tearing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inline-integration</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Index reduction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Automatic differentiation</td>
<td></td>
<td>$1-10k</td>
</tr>
<tr>
<td>Availability</td>
<td>$1-10k</td>
<td>Executable free (kernel is not open-source)</td>
</tr>
</tbody>
</table>
**Partitioning or Component Reduction**

1: \( Q = T_1 - T_2 \)

2: \( 0 = T_1 \)

3: \( f(t) = T_1 + T_2 \)

Application to an electric circuit model

Suppose we have an equation

\[ 0 = f(x), \ x \in \mathbb{R}^n, \ n > 1, \]

that can be written in the form

(1) \quad L \ x^1 = \hat{f}^1(x^2),

(2) \quad 0 = \hat{f}^2(x^1, x^2),

where \( L \) is a lower triangular matrix with constant non-zero diagonals.

Now, pick a guess value for \( x^2 \), solve (1) for \( x^1 \), and compute a new value for \( x^2 \) from (2). Iterate until \( x^2 \) converges to a solution.
SPARK/OpenModelica Integration
SPARK Integration

- Modelica Source Code
- Translator
- Analyzer
- Optimizer
- Code Generator
- C Compiler
- Simulation

Modelica model
Flat Model
Sorted equations
Optimized sorted equations
C Code
Executable

SPARK integration

Match variables with equations
Derive equation inverses
Use SPARK solver
- symbolic processing
- numerical solver

Fritzson et al. (2007)
Integration

- Modify the open source Modelica compiler to
  - Construct internal data structures corresponding to SPARK classes (individual equations and sub-systems)
  - Generate SPARK class definitions
  - Use another open source system (e.g., SAGE) to perform needed algebraic manipulations
- Use Modelica annotations (part of the Modelica standard) to give user access to SPARK controls
- Modify SPARK API to integrate diagnostics into OpenModelica (future work)
- Build algorithmic code generator (near future work)
Current Status

- Compiler now emits basic version of SPARK model
  - No user control via annotation
  - Does not provide code for re-use
- Compiler supports data structures corresponding to SPARK classes (sub-models) including user controls.
- Need to build code generator for the latter (straightforward task)
Dymola/SPARK Benchmarking
Test problem:

- Airflow network (pressure & mass flow rate only)
- Algebraic system of equations
- Scaled VAV air distribution system (ASHRAE 825-RP)
Dymola/SPARK Benchmarking

Schematic view of VAV System described in ASHRAE 825-RP
model Splitter

"Splitting/joining component with static balances for an infinitesimal control volume"

FlowNetwork.Interfaces.FluidPort_b port_1 ∈;
FlowNetwork.Interfaces.FluidPort_b port_2 ∈;
FlowNetwork.Interfaces.FluidPort_b port_3 ∈;

Modelica.SIunits.AbsolutePressure p(nominal=185) "Pressure";

equation

port_1.m_flow + port_2.m_flow + port_3.m_flow = 0 "Mass balance";

// Momentum balances
port_1.p = p;
port_2.p = p;
port_3.p = p;

end Splitter;
Dymola/SPARK Benchmarking
Dymola/SPARK Benchmarking

Symbolic processor

Numerical performance

Table 1: Dimension of biggest nonlinear system of equations. For the SPARK, no ML denotes no MATCH_LEVEL and with ML denotes with MATCH_LEVEL.

<table>
<thead>
<tr>
<th>$N_{sui}$</th>
<th>SPARK</th>
<th>Dymola</th>
<th>$\dot{m} = f(\Delta p)$</th>
<th>$\Delta p = g(\dot{m})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>21</td>
<td>18</td>
<td>41</td>
<td>26</td>
</tr>
<tr>
<td>2</td>
<td>36</td>
<td>34</td>
<td>71</td>
<td>47</td>
</tr>
<tr>
<td>3</td>
<td>51</td>
<td>50</td>
<td>101</td>
<td>66</td>
</tr>
<tr>
<td>4</td>
<td>66</td>
<td>65</td>
<td>131</td>
<td>87</td>
</tr>
<tr>
<td>5</td>
<td>81</td>
<td>80</td>
<td>161</td>
<td>106</td>
</tr>
</tbody>
</table>

![Graph showing CPU time vs. system size](image-url)
Summary of Benchmarks

- SPARK is compatible with Modelica language
- SPARK has a robust, high performance solver for nonlinear equations, competitive with expensive commercial systems
- OpenModelica has provided a useful testbed for integrating SPARK

Future work…
Future Work

**SPARK/OpenModelica:** Continue integration

**Modelica modeling:** Open-source buildings library development

**Modelica co-simulation:** Open source Building Controls Virtual Test Bed development. Will link Modelica to EnergyPlus.

https://gaia.lbl.gov/bir

https://gaia.lbl.gov/bcvtb