Utopia: A High Performance C++ Embedded Domain Specific Language for Scientific Computing

N.A. Fadel1, A. Fink2, P. Zulian3, D. Karvounis3, R. Krause3

1ETHZ/SCS - Swiss National Supercomputing Centre, 2ETHZ/SED - Swiss Seismological Service, 3USI - Università della Svizzera italiana

FASTER Project

More than 75% of the global energy consumption is being produced by technologies that can lead to man-made earthquakes. Although such technologies only on potential dangerous induced seismicity, they can assist in reducing greenhouse emissions. ETH and USI have developed an algorithmic framework for long-term forecasts and assessments both of Seismicity and of Thermal Emissions.

- 3D Monitor Carlo simulations of randomly sampled and dynamically changing reservoirs and solution in a hybrid model (pressure diffusion is computed both inside the 3D continuous reservoir and inside the higher diffusion fracture 3D planes embedded in the reservoir).
- The model contains deterministic 3D-dielectric fracture modeling for flow and heat transport with mechanical modeling of fracture transport, location, source and magnitude of induced seismicity events.

FASTER project goal is accelerate this framework through:

- optimizing code, employing more efficient numerical solvers, simplifying the schematics, and ensuring Phase Field method, and increasing the
- FASTER project g ➔ long term Forecasts and Assessments both of Seismicity and of Thermal Energy, etc.

The resulting software is going to be a part of the Adaptive Traffic Light System

The model combines deterministic 3D discrete fracture modeling for flow and heat transport with mechanical modeling of fracture transport, location, source and magnitude of induced seismicity events.

Mathematical model

\[ \frac{\partial \sigma}{\partial t} + \nabla \cdot ( \sigma \cdot \mathbf{v} ) = - \nabla \cdot \mathbf{q} + f \]

Mass balance equation

\[ \nabla \cdot ( \mathbf{K} \nabla \phi ) = q \]

Mohr-Coulomb failure criterion

\[ |\mathbf{K} \nabla \phi| > C + \mathbf{K} \cdot \mathbf{t} \]

Implementation and Results

<table>
<thead>
<tr>
<th>Trinillos</th>
<th>ML</th>
<th>CPU</th>
<th>GPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total (s)</td>
<td>104</td>
<td>78</td>
<td>52</td>
</tr>
<tr>
<td>Solver</td>
<td>55</td>
<td>34</td>
<td>20</td>
</tr>
<tr>
<td>Setup</td>
<td>7</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>Output</td>
<td>35</td>
<td>29</td>
<td>17</td>
</tr>
<tr>
<td>Speedup</td>
<td>1x</td>
<td>1.33x</td>
<td>2.00x</td>
</tr>
</tbody>
</table>

FASTER Representative testcase: Basel's basin benchmark

Implementation speedup 516x

EDSL: eDSL primitives are realized by exploiting the C++ language meta-programming facilities, function overloading, and functional-style programming constructs introduced in the C++11 standard

Tensor types are represented by the wrapper class

Structured tensorial quantities from a back-end representation.

EDSL is based on:

- eDSL
- expression-tree
- EDSL: Embedded Domain Specific Language for Scientific Computing

Expression tree: The nodes of the tree are expressions and can be either operations, wrapper objects, or factories.

Objectives:

1. Specify compositions of expressions without actually creating concrete tensor objects
2. Creation of objects such as the zero, identity, and sparse matrices

EDSL: eDSL primitives are realized by exploiting the C++ language meta-programming facilities, function overloading, and functional-style programming constructs introduced in the C++11 standard

Interaction between wrappers is defined through the Utopia primitives, that automatically generates an expression tree which is evaluated only when it is assigned to another wrapper object

Evaluation: The purpose of the evaluator component is composing tensorial quantities from a given expression tree by means of back-ends:

1. The back-end provides data-types and algorithms
2. Support expression to function mapping. This strategy is performed by mapping expressions to functions of specific back-ends.

Remarks:

- Write once, use on multiple platforms
- Write once, use both linear algebra libraries (PETSc or Trilinos)
- Lower learning curve compared to PETSc and Trilinos
- Speed up custom code using state of art numerical libraries

References:

- DÖF 86
- Karvounis, Werner, Decision Making Software for Forecasting Induced Seismicity and Thermal Emissions
- Zulian, P., Utopia A C++ Embedded Domain Specific Language for Scientific Computing, Zulian et al., to be published
- The Design of Trinillos, Herrera, Salas, Applied Parallel Computing, 2006, pg 620
- PETSc, Users Manual, Bailey and et al., Argonne National Laboratory, 2018

Future Steps

- Use Utopia for other PASC projects
- eDSL for high performance multi-element assembly
- eDSL for Block Preconditioner combination

In contrast, embedded domain specific languages (eDSLs) use the same language and compiler for both the "scripting" layer and the implementation of the back-end, for this reason, eDSLs have the opportunity to provide the right balance between abstraction and direct access to the back-end data types and algorithms.