Stencil Domain: Multigrid

- Elliptic PDEs and systems thereof
- Discretization using finite differences or volumes
- Patch-based domains
Domain-Specific Stencil Language ExaSlang

Layer 1:
Continuous Domain & Continuous Model

Layer 2:
Discrete Domain & Discrete Model

Layer 3:
Algorithmic Components & Parameters

Layer 4:
Complete Program Specification

abstract problem formulation

concrete solver implementation

Target Platform Description
ExaSlang Layers

• **Layer 1 (continuous)**
  
  Support of Unicode and LaTeX symbols in a continuous problem definition. Optional specification of discretization and solver options used to auto-generate lower layers. Support for automatic finite difference discretization of operators.

• **Layer 2 (discrete)**
  
  Discretized functions are fields (data type, grid location), tied to a domain. Geometric information as “virtual fields”, resolved to constants or field accesses. (Discretized) Operators as stencils or stencil templates.

• **Layer 3 (solver)**
  
  Specification of a solver for the discrete problem, either by hand or set up automatically. Support of a Matlab-like syntax.

• **Layer 4 (application)**
  
  Tuning of communication patterns. Specification of the main application, I/O, performance evaluation and visualization.
Polyhedral Search Space Exploration
Stefan Kronawitter and Christian Lengauer.

Polyhedral Search Space Exploration in the ExaStencils Code Generator.

Stencil Codes
- are memory-bandwidth bound
  - goal: reduce bandwidth requirements, increase cache efficiency
  - spatial blocking: divide computation into smaller parts
  - temporal blocking: combine subsequent smoothing steps

Polyhedron Model
- supports both blocking techniques
- but model-driven approaches do not pay
  ➔ add focus to the optimization

Focussed Optimization
- stencil-unaware: guided schedule exploration
- stencil-aware: seven filters specific to the stencil-domain
- result: in a few seconds a single-digit number of very good schedules
- hope and expectation: there are also filters for other domains
Polyhedral Search Space Exploration

Stefan Kronawitter and Christian Lengauer.

Polyhedral Search Space Exploration in the ExaStencils Code Generator.


Comparison with other Tools and Algorithms

- others are better in some experiments, but worse in others
- exploration results are good for all experiments
- exploration time almost neglectable for high filter level (many filters used)

<table>
<thead>
<tr>
<th></th>
<th>Jacobi 3D</th>
<th></th>
<th></th>
<th>RBGS 3D</th>
<th></th>
<th></th>
<th>Jacobi 2D</th>
<th></th>
<th></th>
<th>RBGS 2D</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>cc1</td>
<td>cc2</td>
<td>ccd</td>
<td>vc1</td>
<td>cc1</td>
<td>vc1</td>
<td>cc1</td>
<td>cc2</td>
<td>ccd</td>
<td>vc1</td>
</tr>
<tr>
<td>baseline</td>
<td>34%</td>
<td>56%</td>
<td>81%</td>
<td>53%</td>
<td>32%</td>
<td>57%</td>
<td>25%</td>
<td>31%</td>
<td>32%</td>
<td>27%</td>
</tr>
<tr>
<td>guided exploration</td>
<td>100%</td>
<td>100%</td>
<td>99%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>83%</td>
<td>89%</td>
<td>90%</td>
<td>83%</td>
</tr>
<tr>
<td>isl</td>
<td>11%</td>
<td>12%</td>
<td>6%</td>
<td>31%</td>
<td>20%</td>
<td>54%</td>
<td>8%</td>
<td>7%</td>
<td>7%</td>
<td>12%</td>
</tr>
<tr>
<td>simple</td>
<td>96%</td>
<td>96%</td>
<td>6%</td>
<td>100%</td>
<td>22%</td>
<td>55%</td>
<td>82%</td>
<td>88%</td>
<td>7%</td>
<td>77%</td>
</tr>
<tr>
<td>heuristics</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLuTo</td>
<td>68%</td>
<td>85%</td>
<td>87%</td>
<td>90%</td>
<td>8%</td>
<td>41%</td>
<td>50%</td>
<td>61%</td>
<td>55%</td>
<td>47%</td>
</tr>
<tr>
<td>rectangular</td>
<td>50%</td>
<td>36%</td>
<td>49%</td>
<td>53%</td>
<td>72%</td>
<td>63%</td>
<td>62%</td>
<td>44%</td>
<td>48%</td>
<td>70%</td>
</tr>
<tr>
<td>unrolled</td>
<td>72%</td>
<td>85%</td>
<td>100%</td>
<td>90%</td>
<td>—</td>
<td>—</td>
<td>75%</td>
<td>83%</td>
<td>87%</td>
<td>86%</td>
</tr>
<tr>
<td>diamond</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PolyMage</td>
<td>49%</td>
<td>57%</td>
<td>70%</td>
<td>47%</td>
<td>31%</td>
<td>44%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>Polyite</td>
<td>34%</td>
<td>50%</td>
<td>53%</td>
<td>64%</td>
<td>—</td>
<td>—</td>
<td>23%</td>
<td>29%</td>
<td>33%</td>
<td>31%</td>
</tr>
</tbody>
</table>
Data Layout Transformations

Stefan Kronawitter, Sebastian Kuckuk, Harald Köstler, and Christian Lengauer.
Automatic Data Layout Transformations in the ExaStencils Code Generator.

Red-Black Gauss-Seidel Kernel

```c
// color splitting
LayoutTransformation {
  transform Solution and RHS
    with [x, y] => [x/2, y, (x+y)%2]
}
```
Data Layout Transformations


Red-Black Gauss-Seidel Kernel

```plaintext
// color splitting
LayoutTransformation {
  transform Solution and RHS
    with [x, y] => [x/2, y, (x+y)%2]
}
```

![Chart showing performance comparisons between CPU and GPU for different data layout transformations.](chart.png)
**Data Layout Transformations**

Stefan Kronawitter, Sebastian Kuckuk, Harald Köstler, and Christian Lengauer.

*Automatic Data Layout Transformations in the ExaStencils Code Generator.*


---

**Optical Flow Simulation**

```plaintext
LayoutTransformations {
    concat @finest Ix, Iy, Iz, It into I
    concat IxIx, IxIy, IxIz, IyIy, IyIz, IzIz into II
    transform I@finest, II@((finest-1) to finest),
        rhs@((finest-1) to finest), flow@((finest-1) to finest)
    with [x,y,z] => [x/2,y,z,(x+y+z)%2]
    transform residual, cgTmp0, cgTmp1, II@(0 to (finest-2)),
        rhs@(0 to (finest-2)), flow@(0 to (finest-2))
    with [x,y,z,v] => [v,x,y,z]
}
```

---

**Data Layout Transformations**

- arbitrary linear transformations supported
- application code need not be modified or prepared, adding a `LayoutTransformations` block is sufficient

---

color splitting:
separate red and black points

SoA to AoS transformation for vector fields and concatenated II
Data Layout Transformations

Stefan Kronawitter, Sebastian Kuckuk, Harald Köstler, and Christian Lengauer.

Automatic Data Layout Transformations in the ExaStencils Code Generator.


Optical Flow Simulation

```
LayoutTransformations {
    concat IxIx, IxIy, IxIz, IyIy, IyIz, IzIz into II
    transform II((finest-1) to finest),
        rhs((finest-1) to finest), flow((finest-1) to finest)
    with [x,y,z] => [x/2,y,z,(x+y+z)mod2]
    transform residual, cgTmp0, cgTmp1, II(0 to (finest-2)),
        rhs(0 to (finest-2)), flow(0 to (finest-2))
    with [x,y,z,v] => [v,x,y,z]
}
```
**Distance-Based Sampling**

- All configurations
- Valid configurations

**Our proposition:** Distance-based sampling enables the selection of a uniformly distributed set of configurations.

**Evaluation:** Our new approach leads to results with higher accuracy than existing approaches.

<table>
<thead>
<tr>
<th></th>
<th>Coverage-based</th>
<th></th>
<th></th>
<th></th>
<th>Solver-based</th>
<th></th>
<th></th>
<th></th>
<th>Randomized solver-based</th>
<th></th>
<th></th>
<th></th>
<th>Diversified distance-based</th>
<th></th>
<th></th>
<th></th>
<th>random</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(t = 1)</td>
<td>(t = 2)</td>
<td>(t = 3)</td>
<td></td>
<td>(t = 1)</td>
<td>(t = 2)</td>
<td>(t = 3)</td>
<td></td>
<td>(t = 1)</td>
<td>(t = 2)</td>
<td>(t = 3)</td>
<td></td>
<td>(t = 1)</td>
<td>(t = 2)</td>
<td>(t = 3)</td>
<td></td>
<td>(t = 1)</td>
<td>(t = 2)</td>
<td>(t = 3)</td>
</tr>
<tr>
<td>7z</td>
<td>54.1%</td>
<td>54.1%</td>
<td>24.4%</td>
<td></td>
<td>78.0%</td>
<td>58.2%</td>
<td>24.3%</td>
<td></td>
<td>64.5%</td>
<td>37.5%</td>
<td>21.8%</td>
<td></td>
<td>78.5%</td>
<td>19.6%</td>
<td>17.3%</td>
<td></td>
<td>58.2%</td>
<td>15.1%</td>
<td>9.9%</td>
</tr>
<tr>
<td>BDB-C</td>
<td>122.9%</td>
<td>29.0%</td>
<td>26.5%</td>
<td></td>
<td>69.5%</td>
<td>66.0%</td>
<td>61.8%</td>
<td></td>
<td>65.0%</td>
<td>64.6%</td>
<td>22.3%</td>
<td></td>
<td>77.2%</td>
<td>45.6%</td>
<td>13.6%</td>
<td></td>
<td>121.3%</td>
<td>39.1%</td>
<td>12.2%</td>
</tr>
<tr>
<td>Dune</td>
<td>19.1%</td>
<td>13.3%</td>
<td>11.2%</td>
<td></td>
<td>23.1%</td>
<td>15.1%</td>
<td>11.9%</td>
<td></td>
<td>39.9%</td>
<td>15.7%</td>
<td>10.9%</td>
<td></td>
<td>17.3%</td>
<td>12.7%</td>
<td>11.5%</td>
<td></td>
<td>17.6%</td>
<td>11.5%</td>
<td>11.3%</td>
</tr>
<tr>
<td>Hipacc</td>
<td>26.2%</td>
<td>20.4%</td>
<td>20.4%</td>
<td></td>
<td>44.5%</td>
<td>16.8%</td>
<td>14.5%</td>
<td></td>
<td>27.7%</td>
<td>14.5%</td>
<td>14.2%</td>
<td></td>
<td>33.9%</td>
<td>13.9%</td>
<td>13.4%</td>
<td></td>
<td>19.9%</td>
<td>13.9%</td>
<td>13.4%</td>
</tr>
<tr>
<td>JavaGC</td>
<td>44.7%</td>
<td>32.1%</td>
<td>23.5%</td>
<td></td>
<td>57.6%</td>
<td>64.5%</td>
<td>36.7%</td>
<td></td>
<td>40.6%</td>
<td>37.9%</td>
<td>32.1%</td>
<td></td>
<td>58.9%</td>
<td>15.4%</td>
<td>13.2%</td>
<td></td>
<td>55.8%</td>
<td>13.9%</td>
<td>12.3%</td>
</tr>
<tr>
<td>LLVM</td>
<td>6.2%</td>
<td>6.2%</td>
<td>5.8%</td>
<td></td>
<td>8.9%</td>
<td>5.5%</td>
<td>5.2%</td>
<td></td>
<td>5.6%</td>
<td>5.2%</td>
<td>5.2%</td>
<td></td>
<td>6.5%</td>
<td>6.1%</td>
<td>5.2%</td>
<td></td>
<td>5.6%</td>
<td>5.2%</td>
<td>5.2%</td>
</tr>
<tr>
<td>lzip</td>
<td>27.2%</td>
<td>16.7%</td>
<td>11.3%</td>
<td></td>
<td>51.9%</td>
<td>26.4%</td>
<td>24.3%</td>
<td></td>
<td>42.2%</td>
<td>20.8%</td>
<td>16.3%</td>
<td></td>
<td>122.2%</td>
<td>26.7%</td>
<td>14.3%</td>
<td></td>
<td>62.7%</td>
<td>18.3%</td>
<td>15.6%</td>
</tr>
<tr>
<td>Polly</td>
<td>18.6%</td>
<td>12.7%</td>
<td>7.4%</td>
<td></td>
<td>20.5%</td>
<td>18.2%</td>
<td>14.9%</td>
<td></td>
<td>20.1%</td>
<td>15.2%</td>
<td>14.9%</td>
<td></td>
<td>27.8%</td>
<td>16.5%</td>
<td>13.7%</td>
<td></td>
<td>25.1%</td>
<td>13.0%</td>
<td>10.3%</td>
</tr>
<tr>
<td>VP9</td>
<td>104.2%</td>
<td>36.5%</td>
<td>36.5%</td>
<td></td>
<td>183.4%</td>
<td>109.1%</td>
<td>48.0%</td>
<td></td>
<td>135.7%</td>
<td>261.3%</td>
<td>94.5%</td>
<td></td>
<td>80.0%</td>
<td>42.1%</td>
<td>32.0%</td>
<td></td>
<td>80.6%</td>
<td>27.2%</td>
<td>23.3%</td>
</tr>
<tr>
<td>x264</td>
<td>16.5%</td>
<td>9.3%</td>
<td>9.0%</td>
<td></td>
<td>25.4%</td>
<td>39.7%</td>
<td>42.6%</td>
<td></td>
<td>18.2%</td>
<td>25.8%</td>
<td>32.9%</td>
<td></td>
<td>13.5%</td>
<td>9.4%</td>
<td>9.1%</td>
<td></td>
<td>13.5%</td>
<td>9.2%</td>
<td>9.1%</td>
</tr>
</tbody>
</table>
A Lightweight, Semi-Automatic Variability Extraction

Approach
- abstraction from the implementation
- extraction of reusable artifacts
- extraction of artifacts used in an application
- identification of alternatives for the artifacts used in the application

Evaluation
- two systems with 8 variation points (e.g., solver type, preconditioner, grid, geometry type, finite element map)
- assessing the quality of the approach by asking the domain expert

Results
- for 5 variation points, all alternatives proposed by the domain expert identified
- for 2 variation points, alternatives identified that did not occur to the domain expert

Cooperation with ExaDune
Genetic Programming for Solver Optimization

- Automatic construction of multigrid solvers for given discretizations
- Multi-objective: convergence rate and execution time per cycle
- Fitness estimation based on LFA and roofline analysis
- Final evaluation of promising individuals through ExaStencils backend
SYCL

- SYCL is a modern frontend to the OpenCL ecosystem
  - Single-source multiple compiler passes principle
  - Task-graph based execution model
  - Automatic data transfers between host and devices

- Access to OpenCL devices
  - "classical" CPUs
  - GPUs
  - FPGAs (upcoming)

- Analogously to OpenCL, SYCL is a specification with different implementations
  - ComputeCpp (CPUs, GPUs; commercial)
  - triSYCL (CPUs, FPGAs (upcoming); open-source)
  - hipSYCL (GPUs; open-source)
  - sycl-gtx (CPUs, GPUs; open-source)
**SYCL**

- Poisson’s equation with constant and variable coefficients
  - Intel i7-6700 vs. NVIDIA GTX 745
  - ComputeCpp: experimental support for NVIDIA GPUs
  - ComputeCpp-generated real OpenCL kernels with memory transfers etc.
  - triSYCL (CPU) uses blockwise OpenMP

- two master theses in 2018
**In-Situ Visualization Capabilities**

Survey of two technologies, both offering

- interfacing via generated boilerplate code and lightweight, DSL-integrated functions
- computational steering capabilities

Geometry of water distribution

- Custom application built on BGFX
  (M. Obereisenbuchner)

- Popular tool VisIt
  (R. Angersbach)
Appeared / Accepted in 2018  (red = top-rate)

### Workshop, Chapter, Conference:

### Journals:
Honors

Professorships
- Harald Köstler: W2, Friedrich-Schiller Universität, Jena (rejected)
- Sven Apel: W3, Universität des Saarlandes (accepted)

Professional Societies
- Jürgen Teich: Fellow of the IEEE;
  Member of acatech
- Sven Apel: ACM Distinguished Member

Awards
- Hannah Rittich: Verein zur Förderung von Mathematik und Naturwissenschaften e.V.,
  Ph.D. Thesis Award
- Sebastian Schweikl: SPPEXA B.Sc. Award
- Sebastian Kuckuk: CoSaS 2018 Best Poster Award
- Sven Apel: ASE Distinguished Reviewer Award

Degrees / Titles
- Frank Hannig: Habilitation, Friedrich-Alexander Universität Erlangen-Nürnberg
- Harald Köstler: Professor, Friedrich-Alexander Universität Erlangen-Nürnberg