Polly-ACC: Transparent Compilation to Heterogeneous Hardware
Tobias Grosser, Torsten Hoefler
Sequential Software

Fortran

C/C++

Parallel Hardware

Multi-Core CPU

Accelerator
Design Goals

Automatic

Non-Goal: Algorithmic Changes

“Regression Free”

High Performance
Tool: Polyhedral Modeling

Program Code

```c
for (i = 0; i <= N; i++)
for (j = 0; j <= i; j++)
S(i,j);
```

Iteration Space

\[ D = \{ (i,j) \mid 0 \leq i \leq N \land 0 \leq j \leq i \} \]

- \( N = 4 \)
- \( (i, j) = (4, 4) \)

Polly -- Performing Polyhedral Optimizations on a Low-Level Intermediate Representation
Tobias Grosser et al, Parallel Processing Letter, 2012
Mapping Computation to Device

**Iteration Space**

\[ BID = \{(i,j) \rightarrow (\left\lfloor \frac{i}{4} \right\rfloor \mod 2, \left\lfloor \frac{j}{3} \right\rfloor \mod 2)\} \]

\[ TID = \{(i,j) \rightarrow (i \mod 4, j \mod 3)\} \]
Memory Hierarchy of a Heterogeneous System

- Main Memory
- Device Memory
- Shared Memory
- Registers

CPU  CPU
CPU  CPU
Host-device date transfers
Host-device date transfers

Main Memory

Device Memory

Shared Memory

Registers

CPU

GPU

CPU

GPU

CPU

GPU

CPU

GPU
Mapping onto fast memory
Mapping onto fast memory

Polyhedral parallel code generation for CUDA, Verdoollaeg, Sven et. al, ACM Transactions on Architecture and Code Optimization, 2013
Profitability Heuristic

- All Loop Nests
- Trivial
- Unsuitable
- Insufficient Compute
- Static
- Dynamic
- Execution
- GPU
- Modeling

ETH Zürich
From kernels to program – data transfers

```c
void heat(int n, float A[n], float hot, float cold) {
    float B[n] = {0};
    initialize(n, A, cold);
    setCenter(n, A, hot, n/4);
    for (int t = 0; t < T; t++) {
        average(n, A, B);
        average(n, B, A);
        printf("Iteration %d done", t);
    }
}
```
Data Transfer – Per Kernel

Host Memory  Device Memory

initialize()  \( D \rightarrow H \)

setCenter()  \( D \rightarrow H \)

average()  \( H \rightarrow D \)  \( D \rightarrow H \)

average()  \( H \rightarrow D \)  \( D \rightarrow H \)

average()  \( H \rightarrow D \)  \( D \rightarrow H \)
Data Transfer – Inter Kernel Caching

Host Memory

Device Memory

- initialize()
- setCenter()
- average()
- average()
- average()

$D \rightarrow HH \rightarrow D$
Evaluation

Workstation: 10 core SandyBridge
Mobile: 4 core Haswell
NVIDIA Titan Black (Kepler)
NVIDIA GT730M (Kepler)
LLVM Nightly Test Suite

# Compute Regions / Kernels

<table>
<thead>
<tr>
<th></th>
<th>No Heuristics</th>
<th>Heuristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCoPs</td>
<td>16</td>
<td>100</td>
</tr>
<tr>
<td>0-dim</td>
<td>1000</td>
<td>100</td>
</tr>
<tr>
<td>1-dim</td>
<td>10000</td>
<td>100</td>
</tr>
<tr>
<td>2-dim</td>
<td>10000</td>
<td>100</td>
</tr>
<tr>
<td>3-dim</td>
<td>1000</td>
<td>10</td>
</tr>
</tbody>
</table>
Polybench 3.2

Baseline: icc –O3 (sequential), 10 core CPU + NVIDIA Titan Black (workstation)
Lattice Boltzmann (SPEC 2006)
Cactus ADM (SPEC 2006)

Performance [iterations/second]

Mobile

Report output interval

Workstation

Report output interval

Polly ACC cached
Polly ACC
icc parallel
icc
LLVM

0.6x
0.5x

1.9x
1.9x

2.1x
2.3x

12.9x
14.1x
Cactus ADM (SPEC 2006) - Data Transfer

**Mobile**

- H to D
- H to D (cached)
- D to H
- D to H (cached)

**Workstation**

- H to D
- H to D (cached)
- D to H
- D to H (cached)

Time used for data transfers [s] vs. Report output interval.
Polly-ACC

Mapping Computation to Device

Data Transfer – Per Kernel

Profitability Heuristic

"Regression Free"

High Performance

Automatic

BID = \{(i,j) \rightarrow (\left\lfloor \frac{i}{1} \right\rfloor \% 2, \left\lfloor \frac{j}{3} \right\rfloor \% 2)\}

TID = \{(i,j) \rightarrow (i \% 4,j \% 3)\}

http://spcl.inf.ethz.ch/Polly-ACC