NARMADA: Near-memory horizontal diffusion accelerator for scalable stencil computation

Gagandeep Singh, Dionysios Diamantopoulos, Sander Stuijk, Christoph Hagleitner, and Henk Corporaal

sin@zurich.ibm.com

FPL, Barcelona
11th-September-2019
Stencil Computations and Applications

Stencils are used in ~30% of HPC applications:

- Fluid dynamics, image processing, atmospheric modelling

Workload Characteristics

High-order stencil computations are cache unfriendly
Workload Characteristics

High-order stencil computations are cache unfriendly

• Limited arithmetic intensity: only reuse potential in neighboring pixels
• Sparse and complex access pattern
Workload Characteristics

High-order stencil computations are cache unfriendly

- Limited arithmetic intensity: only reuse potential in neighboring pixels
- Sparse and complex access pattern

7-point Jacobi in 3D plane

Image source: Performance Tuning and Analysis for Stencil-Based Applications on POWER8 Processor – Xu et al. (2018)
Workload Characteristics

High-order stencil computations are cache unfriendly

- Limited arithmetic intensity: only reuse potential in neighboring pixels

Data movement bottleneck

Image source: Performance Tuning and Analysis for Stencil-Based Applications on POWER8 Processor – Xu et al. (2018)
Stencil use in COSMO

Stencil computing in weather/climate applications
  • Dynamical core is the most essential part of the weather models
Stencil use in COSMO

Stencil computing in weather/climate applications

- Dynamical core is the most essential part of the weather models
- \( O(100) \) different stencil compute motifs
- \( \sim 30 \) variable- and \( \sim 70 \) temporary arrays (3D grids)

Section of COSMO CDAG
(Courtesy CSCS/ETHz and Ronald Luijten)
Stencil use in COSMO

Stencil computing in weather/climate applications
- Dynamical core is the most essential part of the weather models
- $O(100)$ different stencil compute motifs
- $\sim 30$ variable- and $\sim 70$ temporary arrays (3D grids)
- Complex stencil programs

Section of COSMO CDAG
(Courtesy CSCS/ETHZ and Ronald Luijten)
Stencil use in COSMO

Stencil computing in weather/climate applications
• Dynamical core is the most essential part of the weather models
• ~30 variable and ~70 temporary arrays (3D grids)
• Complex stencil programs

Not another “elementary” stencil talk!
Horizontal Diffusion ("complex" stencil)

- Compound stencil kernel, consists of a collection of elementary stencil kernels
Horizontal Diffusion ("complex" stencil)

- Compound stencil kernel, consists of a collection of elementary stencil kernels
- Iterates over 3D grid that perform laplacian and flux operations
- Complex memory access behavior and low arithmetic intensity

Stencil composition (Courtesy CSCS/ETHz and Ronald Luijten)
Horizontal Diffusion (“complex” stencil)

- Compound stencil kernel, consists of a collection of elementary stencil kernels
- Iterates over 3D grid that perform laplacian and flux operations
- Complex memory access behavior and low arithmetic intensity

T. Gysi, T. Grosser, T. Hoefler: MODESTO: Data-centric Analytic Optimization of Complex Stencil Programs on Heterogeneous Architectures, ICS’15
DFG from T. Hoefler
IBM POWER9 CPU Roofline

POWER9 Roofline (8335-GTH, 16-cores, SMT4)

972.8 GFLOP/s/socket (3.8GHz x 16cores x 16flops/cycle)

Performance gap due to a combination of memory-hierarchy under-utilization and application's complex access patterns

Optimized hdiff for P9 - 64 threads
Optimized hdiff for P9 - 1 thread
Arithmetic Intensity for hdiff

Optimizable bandwidth (SW)
110GBps experimental BW on STREAM
13-cache (208.57)x16=3337GBps
IBM POWER9 CPU Roofline

POWER9 Roofline (8335-GTH, 16-cores, SMT4)

972.8 GFLOP/s/socket (3.8GHz x 16cores x 16flops/cycle)

Performance gap due to a combination of memory-hierarchy under-utilization and application's complex access patterns

- L3-cache (208.57)x16=3337GBps
- DRAM
- 110GBps experimental BW on STREAM
- 59.6GFlop/s
- 5.1GFlop/s

- Optimized hdiff for P9 - 64 threads
- Optimized hdiff for P9 - 1 thread
- Arithmetic Intensity for hdiff
Alternative Platforms

FPGAs ideal for adapting to rapidly evolving workloads!
Heterogeneous Architecture: CPU+FPGA

- Host System
  IBM POWER9-16 core (64-threads)
  Power: IBM AMESTER\(^1\)

- FPGA board
  Xilinx Virtex\(^\circledR\) Ultrascale+\(^{\text{TM}}\) XCVU3P-2

\(^1\)https://github.com/open-power/amester
Traditional I/O Technology

3 versions of the data (not coherent).
1000s of instructions in the device driver.

Typical I/O Model Flow: Total ~13µs for data prep

- DD Call
- Copy or Pin Source Data
- MMIO Notify Accelerator
- Acceleration
- Poll / Interrupt Completion
- Copy or Unpin Result Data
- Ret. From DD Completion

300 instructions: 7.9µs
10,000 instructions: 4.9µs
3,000 instructions: 1,000 instructions: 1,000 instructions
CAPI Technology Overview

Flow with a CAPI Model:

- Shared Mem. Notify Accelerator: 400 Instructions, 0.3μs
- Acceleration (Dependent, but Equal to above): 100 Instructions, 0.06μs
- Total 0.36μs

Memory Subsystem

1 coherent version of the data. No device driver call/instructions.
Accelerator Framework

- Accelerators are acting as peers to CPU
- High-performance cache-coherent link
- An interrupt-based queuing mechanism
- Minimal CPU usage (thus power) during FPGA use

https://github.com/open-power/snap
## Dataflow Sequence

1. **1 POWER9 Cache-line**
   - 1024 bits = 128B -> 32 x float32

2. **FPGA AXI Register**
   - 512 bits x 2 reads

3. **FPGA Cacheline Buffer**
   - 32 x float32

   - **Software-defined Host-to-FPGA Data Unpacking**
     - 32 addresses
     - 1 x float32 -> 32 bits

4. **Cacheline Buffering**

5. **3D Window Gridding**

6. **FPGA DRAM**

7. **2D Vertical Slicing**

8. **BRAM or URAM**

9. **caching for spatial locality**

10. **Register File**

### Diagram Description

- **3D Window**
- **Data in Host DRAM**

©MeteoSwiss

---

[Diagram showing the sequence of dataflow steps and FPGA architecture.]
Dataflow Sequence

1 POWER9 Cache-line
1024bits = 128B -> 32 x float32

FPGA AXI Register
512bits x 2 reads

FPGA Cacheline Buffer
32 x float32

Virtual Address Translation & Coherency by PSL

Software-defined Host-to-FPGA Data Unpacking

32 addresses
1 x float32 -> 32bits

Cacheline Buffering

3D Window Gridding
FPGA DRAM

2D Vertical Slicing
BRAM or URAM

Register File

caching for spatial locality

3D Window

Data in Host DRAM

©MeteoSwiss
Dataflow Sequence

1. **1 POWER9 Cache-line**
   - 1024 bits = 128B -> 32 x float32

2. **FPGA AXI Register**
   - 512 bits x 2 reads

3. **FPGA Cacheline Buffer**
   - 32 x float32

   Software-defined Host-to-FPGA Data Unpacking

4. **3D Window Gridding**

5. **FPGA DRAM**

6. **2D Vertical Slicing**
   - BRAM or URAM

7. **3D Window**

   Data in Host DRAM

8. **on-chip FF**

   caching for spatial locality

9. **Register File**

   ©MeteoSwiss
Dataflow Sequence

1. 1 POWER9 Cache-line
   1024 bits = 128B -> 32 x float32

2. FPGA AXI Register
   512 bits x 2 reads

3. FPGA Cacheline Buffer
   32 x float32

Software-defined Host-to-FPGA Data Unpacking

4. 3D Window Gridding
   FPGA DRAM

5. 2D Vertical Slicing
   BRAM or URAM

6. Caching for spatial locality
   Register File
Dataflow Sequence

1. **1 POWER9 Cache-line**
   1024bits = 128B -> 32 x float32

2. **FPGA AXI Register**
   512bits x 2 reads

3. **FPGA Cacheline Buffer**
   32 x float32

   Software-defined Host-to-FPGA Data Unpacking

   32 addresses

   1 x float32 -> 32bits

4. **Cacheline Buffering**

5. **3D Window Gridding**

6. **FPGA DRAM**

7. **2D Vertical Slicing**

   BRAM or URAM

8. **caching for spatial locality**

   Register File

©MeteoSwiss
Dataflow Sequence

1. **POWER9 Cache-line**
   - 1024 bits = 128B -> 32 x float32

2. **FPGA AXI Register**
   - 512 bits x 2 reads

3. **Software-defined Host-to-FPGA Data Unpacking**
   - 32 addresses
   - 1 x float32 -> 32 bits

4. **Cacheline Buffering**
   - 32 addresses

5. **3D Window Gridding**
   - FPGA DRAM

6. **2D Vertical Slicing**
   - BRAM or URAM

7. **caching for spatial locality**
   - Register File

8. **Virtual Address Translation & Coherency by PSL**

© MeteoSwiss

11
AFU Performance

POWER9 socket (64-threads)
- 8x8x8 window
- 16x16x16 window
- 32x32x32 window

Performance (GFLOP/s)

POWER9 1-thread
- Base: 0.8 0.8 0.8
- Pipeline: 5.6 5.7 5.7
- Partitioned: 11.4 12.4 12.5
- B-Hierarchy: 27.1 32.2 27.2
- U-Hierarchy: 15.6 14.3 13.9

6.3x
AFU Performance

6.3x compared to 1-thread POWER9
AFU Scaling Analysis

![AFU Scaling Analysis Diagram](image-url)
AFU Scaling Analysis

![Graph showing performance (GFLOP/s) across different window sizes and thread counts for POWER9 socket and POWER9 1-thread configurations in B-Hierarchy and U-Hierarchy Scaling.]
AFU Scaling Analysis

3.3x with 18x energy efficiency compared to 1-node POWER9 (16 core)
Accelerator Scaling Prediction

• **Goal:** Quick prediction for scaling AFUs on different FPGA boards

• Long run-time

• Back-of-the-envelope calculations cannot accurately predict complex design behavior

• Heuristics

• **Approach:** Empirical best-fit model using data collected from one device and predict for all the devices in an FPGA family
Accelerator Scaling Prediction

• Xilinx Ultrascale and Ultrascale+ families
  • CAPI enabled
  • With and without URAM
Accelerator Scaling Prediction

- Xilinx Ultrascale and Ultrascale+ families
  - CAPI enabled
  - With and without URAM

- MRE under 15.2%
Accelerator Scaling Prediction

• Xilinx Ultrascale and Ultrascale+ families
  • CAPI enabled
  • With and without URAM

• MRE under 15.2%

• URAM offers more heterogeneity and energy-efficiency with scaling
Future work

• Intra-node multi-FPGA scaling
• Implementation with OpenCAPI and HBM
• General purpose COSMO accelerator
• Trans-precision analysis
• Run-time adaptability for other stencils
Executive Summary

• **Motivation:** Stencil computation is essential part of HPC applications

• **Problem:** Limited performance on conventional architectures

• **Goal:** Study the applicability of compound stencils from real-world weather prediction application on reconfigurable architectures

• **Our contribution:** NARMADA
  • First implementation and optimization of horizontal diffusion kernel from COSMO application on modern heterogeneous system
  • A data-centric heterogeneous memory hierarchy caching scheme with scalability analysis

• **Results**
  • NARMADA has 6.3x performance compared to a 1-thread performance of the state-of-the-art IBM POWER9 CPU
  • 3.3x performance with 18x energy-efficiency compared to a complete IBM POWER9 node
NARMADA: Near-memory horizontal diffusion accelerator for scalable stencil computation

Gagandeep Singh, Dionysios Diamantopoulos, Sander Stuijk, Christoph Hagleitner, and Henk Corporaal

sin@zurich.ibm.com

FPL, Barcelona
11th-September-2019