Application Analysis using Omniperf

Alessandro Fanfarillo, Thomas Gibson
Background: AMD Profiling Tools

**ROC Profiler**
- [github.com/ROCm-Developer-Tools/rocprofiler](https://github.com/ROCm-Developer-Tools/rocprofiler)
- Raw collection of GPU counters and traces
  - Counter collection driven by user-provided input files
- Counter collection output in CSV
- Supported tracing collection
  - HIP
  - HSA
  - GPU
- Traces can be visualized with Perfetto

**Omnitrace**
- [github.com/AMDResearch/omnitrace](https://github.com/AMDResearch/omnitrace)
- Comprehensive trace collection and visualization of CPU + GPU activity
- Support for
  - HIP, HSA, GPU
  - OpenMP®
  - MPI
  - Kokkos
  - Pthreads
  - Multi-GPU
- Traces can be visualized with Perfetto

**Omniperf**
- [github.com/AMDResearch/omniperf](https://github.com/AMDResearch/omniperf)
- Comprehensive collection and visualization of performance counters
- Support for
  - GPU Speed-of-Light Analysis
  - Empirical Roofline Analysis
  - Memory Chart Analysis
  - Kernel Comparisons
- Can be visualized with Grafana
Background: AMD Profiling Tools

<table>
<thead>
<tr>
<th>Objective</th>
<th>How well am I using the GPU?</th>
<th>Why am I seeing this perf?</th>
<th>Where should I focus my time?</th>
</tr>
</thead>
</table>

Analysis Approach
- Roofline
- HW Counters
- Timeline Trace

Focus of this talk

AMD Tools
- Omniperp
- Omnitrace
What is Omniperf?

- Open-source repo
  - github.com/AMDResearch/omniperf
- Built on top of ROC Profiler
  - Drives PMC collection
- Integrated Performance Analyzer for AMD GPUs
  - Roofline Analyzer
  - Mem. Chart Analyzer
  - Speed-of-Light
  - Kernel Comparisons
  - Flexible Filtering/Normalization
  - Comprehensive Profiling:
    - Wavefront Dispatching
    - Shader Compute
    - LDS Accesses
    - L1/L2 Cache Accesses
    - HBM Accesses
- User-interfaces
  - Grafana™ Based GUI
  - Standalone GUI
Roofline Analysis on MI200 GPUs

- Peak MFMA GFLOP/sec
- Peak VALU GFLOP/sec
- Peak L2 BW
- Peak LDS BW
- Peak vL1D BW
- Peak HBM BW
- Workload Perf: (GFLOP/sec, AI)
Roofline Analysis on MI200 GPUs: roofline benchmarking

- Empirical roofline benchmarking
- Measure achievable peak FLOPS
  - VALU: F32, F64
  - MFMA: F16, F32, BF16, F32, F64
- Measure achievable peak BW
  - LDS
  - Vector L1D Cache
  - L2 Cache
  - HBM

- Internally developed micro-benchmark algorithms
  - Peak VALU FLOP: axpy
  - Peak MFMA FLOP: Matrix multiplication based on MFMA intrinsic
  - Peak LDS/vL1D/L2 BW: Pointer chasing
  - Peak HBM BW: Streaming copy

- Note: empirical benchmark runs may not happen if Omniperf version not compatible with installed ROCm version
Getting Started with Omniperf
Getting started with Omniperf: Client

• Step 0: Install ROCm™ 5.1+
  • Fresh installation: Introduction to ROCm Installation Guide for Linux® (amd.com)

• Step 1: Clone Omniperf

```bash
git clone https://github.com/AMDResearch/omniperf.git
```

• Step 2: Install dependencies (requires CMake 3.19+ and Python 3.7+)

```bash
$ cd omniperf
$ export PATH=/path/to/cmake/bin:$PATH
$ export INSTALL_DIR=/path/to/target-installation-dir/omniperf
$ python3 -m pip install --system -- $INSTALL_DIR/python-libs -r requirements.txt
```

• Step 3: Build and install Omniperf client

```bash
$ cd build
$ cmake -DCMAKE_INSTALL_PREFIX=$INSTALL_DIR/1.0.4 -DPYTHON_DEPS=$INSTALL_DIR/python-libs
-DMOD_INSTALL_PATH=$INSTALL_DIR/modulefiles ..
$ make install
```
Getting started with Omniperf: Client

• Step 5: Sanity Check

```
$export PATH=$INSTALL_DIR/1.0.4/bin:$PATH
$export PYTHONPATH=$INSTALL_DIR/python-libs
$export ROOFLINE_BIN=/path/to/installed/omniperf/roofline
$omniperf --version
```

Omniperf version: 1.0.4 (release)
Git revision: 065b4b7

• Online Documentation for Omniperf installation:
  • [amdresearch.github.io/omniperf/installation.html#client-side-installation](http://amdresearch.github.io/omniperf/installation.html#client-side-installation)
  • Includes instructions for systems using modulefiles

---

**Client-side Installation**

Omniperf requires the following basic software dependencies prior to usage:

- Python (>=3.7)
- CMake (>= 3.19)
- ROCm (>= 5.1)

In addition, Omniperf leverages a number of Python packages that are documented in the top-level requirements.txt file. These must be installed prior to Omniperf configuration.

The recommended procedure for Omniperf usage is to install into a shared file system so that multiple users can access the final installation. The following steps illustrate how to install the necessary Python dependencies using pip and Omniperf into a shared location controlled by the INSTALL_DIR environment variable.

**Configuration variables**

The following installation example leverages several CMake project variables defined as follows:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMAKE_INSTALL_PREFIX</td>
<td>controls install path for Omniperf files</td>
</tr>
<tr>
<td>PYTHON_DEPS</td>
<td>provides optional path to resolve Python package dependencies</td>
</tr>
<tr>
<td>MOD_INSTALL_PATH</td>
<td>provides optional path for separate Omniperf modulefile installation</td>
</tr>
</tbody>
</table>
Getting started with Omniperf: Server

Can be installed locally on your workstation. Profile remotely, visualize locally

- **Step 1: setup persistent Docker® storage**

  ```bash
  $ sudo mkdir -p /usr/local/persist
  $ cd /usr/local/persist/
  $ sudo mkdir -p grafana-storage mongodb
  $ sudo docker volume create --driver local --opt type=none --opt device=/usr/local/persist/grafana-storage --opt o=bind grafana-storage
  $ sudo docker volume create --driver local --opt type=none --opt device=/usr/local/persist/mongodb --opt o=bind grafana-mongo-db
  ```

- **Step 2: start the Omniperf server**

  ```bash
  $ sudo docker-compose build
  $ sudo docker-compose up -d
  ```

- **Step 3: configure server**
  - Refer to: [https://amdresearch.github.io/omniperf/](https://amdresearch.github.io/omniperf/)
Omniperf Demonstration
Omniperf: “Hello World” Example

• Step 1: Compile workload

```bash
mkdir test && cd test
$cp $OMNIPERF_HOME/sample/vcopy.cpp .
$hipcc vcopy.cpp -o vcopy
./vcopy 1048576 256
Finished allocating vectors on the CPU
Finished allocating vectors on the GPU
Finished copying vectors to the GPU
sw thinks it moved 1.000000 KB per wave
Total threads: 1048576, Grid Size: 4096 block Size:256, Wavefronts:16384:
Launching the kernel on the GPU
Finished executing kernel
Finished copying the output vector from the GPU to the CPU
Releasing GPU memory
Releasing CPU memory
```

• Step 2: Profile workload

```bash
$omniperf profile -n vcopy_demo -- ./vcopy 1048576 256
```
Omniperf: Live Demonstration on HPC examples

- axpy kernel
- Thread divergence
DISCLAIMERS AND ATTRIBUTIONS

The information contained herein is for informational purposes only and is subject to change without notice. While every precaution has been taken in the preparation of this document, it may contain technical inaccuracies, omissions and typographical errors, and AMD is under no obligation to update or otherwise correct this information. Advanced Micro Devices, Inc. makes no representations or warranties with respect to the accuracy or completeness of the contents of this document, and assumes no liability of any kind, including the implied warranties of noninfringement, merchantability or fitness for particular purposes, with respect to the operation or use of AMD hardware, software or other products described herein. No license, including implied or arising by estoppel, to any intellectual property rights is granted by this document. Terms and limitations applicable to the purchase or use of AMD’s products are as set forth in a signed agreement between the parties or in AMD’s Standard Terms and Conditions of Sale. GD-18

THIS INFORMATION IS PROVIDED "AS IS." AMD MAKES NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE CONTENTS HEREOF AND ASSUMES NO RESPONSIBILITY FOR ANY INACCURACIES, ERRORS, OR OMISSIONS THAT MAY APPEAR IN THIS INFORMATION. AMD SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL AMD BE LIABLE TO ANY PERSON FOR ANY RELIANCE, DIRECT, INDIRECT, SPECIAL, OR OTHER CONSEQUENTIAL DAMAGES ARISING FROM THE USE OF ANY INFORMATION CONTAINED HEREIN, EVEN IF AMD IS EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

© 2023 Advanced Micro Devices, Inc. All rights reserved.

AMD, the AMD Arrow logo, Radeon, Instinct, EPYC, Infinity Fabric, ROCm, and combinations thereof are trademarks of Advanced Micro Devices, Inc. Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies. Docker® and the Docker® logo are trademarks or registered trademarks of Docker, Inc.

Git and the Git logo are either registered trademarks or trademarks of Software Freedom Conservancy, Inc., corporate home of the Git Project, in the United States and/or other countries.

Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.

MongoDB® is a registered trademark of MongoDB, Inc.

Ubuntu® and the Ubuntu® logo are registered trademarks of Canonical Ltd.

Canonical® and the Canonical® logo are registered trademarks of Canonical Ltd.

The OpenMP® name and OpenMP® logo are registered trademarks of the OpenMP Architecture Review Board.