Lab 4 - Roofline Analysis

Advanced Architectures

University of Minho

The Lab 4 focus on the creation of a Roofline model of a server for a sample application using the Intel Advisor software suite. Use a cluster node (e.g., `qsub -qmei -lnodes=1:ppn=4 -lwalltime=...`) to run the analysis through the command line.

This lab tutorial includes four exercises, two to be solved at home (HW 4.x) and two to be solved during the lab class (Lab 4.x).

The goal of Lab 4 is to evaluate the maximum computing throughput and memory bandwidth of a cluster node, while identifying the relevant ceilings that may impact performance of a sample application. The performance data gathered from an application allows it to be plotted into a Roofline graph for a given cluster node, in order to assess its level of optimisation. The relative position of an application operational intensity relatively to the cluster node roofs and ceilings gives an insight into the inefficiencies of the code, which can be better identified and optimised.

To load the compiler in the environment use these commands:

GNU Compiler (for several system libraries): `module load gcc/4.9.0`

Intel Compiler: `source /share/apps/intel/parallel_studio_xe_2019/compilers_and_libraries_2019/linux/bin/compilervars.sh intel64`

Intel Advisor update 5: `source /share/apps/intel/advisor_2019.5.0.602216/advixe-vars.sh` **or**

Intel Advisor update 4: `source /share/apps/intel/advisor_2019.4.0.597843/advixe-vars.sh`

4.1 Installing and using the Intel Advisor

Goals: to get acquainted with the Intel Advisor interface and the Roofline model.

Consider the code provided in the attached file. There are 3 different groups of algorithms, each with a different arithmetic intensity, and thus different bottlenecks.

HW 4.1 The SeARCH cluster provides the updated 4 and 5 of the 2019 Intel Advisor, but only one of these should be used to survey a given computing node (i.e., to run a set of benchmarks to build the Roofline model) and to profile the code and associated dataset.

Download either the 2019 Intel Advisor update 4 or 5 and install it on your laptop (update 5 may not work on OS X). The same update version must be used on the cluster and on your laptop. This installation will be used to visualise the Roofline created for the cluster node.

Create a student account and download the Intel Advisor in the following site:

```
https://software.intel.com/en-us/advisor
```

You must create an account to get a license that is required to use this software. Confirm that your installation is working properly.

HW 4.2 Access the SeARCH cluster and copy the attached file into your home directory. Enable all algorithms in Group 1 in the code and compile it. Submit a job to a compute-6XX node to create the Roofline model containing the dots with the adequate arithmetic intensity for the compiled application:

```
advixe-cl -collect roofline -stack -project-dir /your/dir/roofline_project
- - /your/dir/bin/roofline_demo
```

Copy the `roofline_project` to your laptop and open the project file with your own installation of the Intel Advisor. When executing this software open the *e000* menu, the *Survey & Roofline* tab and then press the *Roofline* vertical tab. A Roofline plot should be displayed inside a window of Intel Advisor.

Answer the following questions:

1. What represent the size and colour of each dot in the Roofline chart dot?
2. What represent the diagonal lines in the Roofline chart? Where should be positioned a dot representing a loop that misses L1 cache too often but hits L2 cache?
3. What represent the horizontal lines in the Roofline chart? Where should be positioned a dot representing a loop that is not vectorized?
4. Where should be positioned a dot that offers a better opportunity for performance improvement?
5. Where are the limiting ceilings for the 3 different data structures (AoS, SoA, and SoA with vectorization) used by the algorithms in Group 1?

Lab 4.1 The compiled algorithms in Group 2 have a higher arithmetic intensity. Analyse these algorithms, similarly to **HW 4.2**.

Based on the data gathered for the algorithms in Group 1 and 2 answer the following questions:

1. Does the displayed Roofline show the peak performance of this cluster node?

2. Which displayed ceilings are not relevant to assess the performance limitations of these algorithms? Remove them from the Roofline visualisation.

3. What is the key optimisation on G1_SOA_SCALAR that is not present in G1_AOS_SCALAR? Does this optimisation allow G1_SOA_SCALAR to break a ceiling limiting the performance of G1_AOS_SCALAR?

4. What is the current limitation of the G1_SOA_VECTOR algorithm? Is it possible to find clues in Intel Advisor that point to this limitation?

5. Can the performance of G1_SOA_VECTOR be further improved? What would be the maximum theoretical performance (in GFLOP/s) that this code could achieve?

6. Consider the algorithms in Group 2. These algorithms have different levels of optimisation, which have a direct impact on their performance. Why is their arithmetic intensity the same?

7. What is the ceiling limiting the performance of G2_AOS_VECTOR? Is it the same as G2_SOA_VECTOR? Does Intel Advisor provide clues to support your claims?

8. What is the vectorization efficiency of G2_AOS_VECTOR? Why is this value so low?

9. Explain the key optimisation implemented on the fastest algorithm of Group 2. How does this optimisation affect the code execution?

10. How much slower is the best performing algorithm relatively to the peak theoretical performance?

11. Modify the code to compute with *float* variables instead of *double*. Is the resulting performance doubled? If not, justify and eventually improve the code.