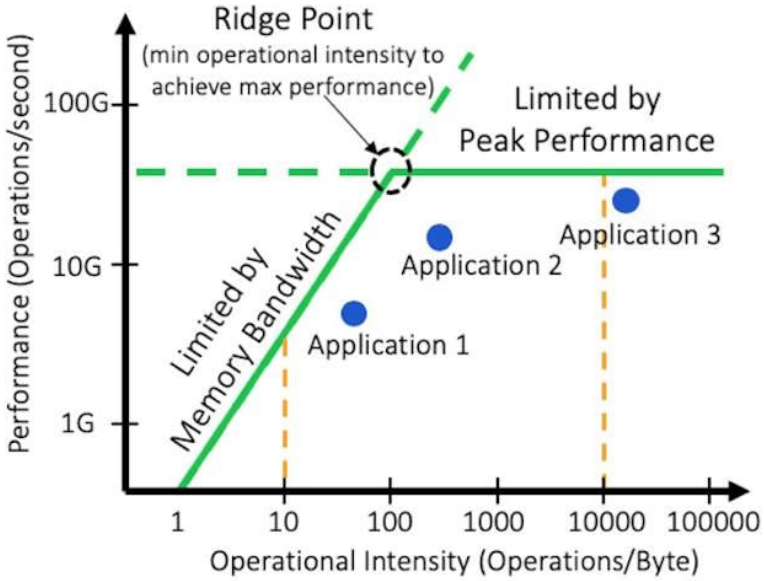
Roofline Model

In this assignment, you will learn how to identify the performance bottleneck in your application (kernel) through roofline analysis.

What is Roofline model?

Roofline modeling is an approach to determine the maximum achievable performance of a kernel given a hardware system configuration. There are broadly two regions in a roofline plot: the memory-bound region and the compute-bound region, and the regions intersect at a single point called the ridge point. The modeling provides a way to analyze if an application, kernel, or a part of code is memory bound (Operational Intensity falls to the left of the ridge-point) or compute bound (Operational Intensity falls to the right of the ridge-point), i.e. are you limit by the operations/sec performance of the compute resources in your system or are the compute units not getting enough data to operate on because of low bytes/sec?



Plotting the peak memory performance:
Since X-axis is OPs per byte and the Y-axis is OPs per second, bytes per second—which equals (Ops/second)/(Ops/byte)—is just a line at a 45-degree angle in the roofline plot.

The ridge point is

$$\left(\frac{Peak\ compute\ performance}{Peak\ memory\ bandwidth}, Peak\ compute\ performance\right)$$

First Article: Roofline: An Insightful Visual Performance Model for Multicore Architectures
You can also look for some technical blogs/articles/videos available online to understand this better.

Your tasks

For this assignment, you will

1. use Intel Advisor to report the performance (INTOPS/sec) of some kernels with different optimizations
    a. The installation steps for Intel Advisor are provided here
    b. The source codes for the kernels are provided here. You will get ten executables (just do *make all*)
2. reason about the movement of application dot in the roofline plot as you optimize the matrix multiplication kernel
    a. See how to use Intel Advisor here
3. summarize the system configuration of the machine you run the kernel on
    a. run ***assignment_roofline/get_config.sh*** on your machine
    b. get L1, L2, L3, and DRAM bandwidth from Intel Advisor
4. answer a numerical question

This assignment is worth 10 points. Please see the breakdown below.
The assignment submission is due on 6:00 pm ET on Jan 27, 2025.

The submission should contain a report having

1. [4 points] the final screenshots of the roofline plot containing application dots for all the ten executables (something like this)
    a. One screenshot comparing 0_vecadd, 0_gemv, 0_matmul
    b. One screenshot comparing 0_matmul, 1_matmul, 2_matmul, 3_matmul, 4_matmul, 4_matmul-sse, 4_matmul-avx, 4_matmul-avx2
2. [2 point] a table showing INTOPS/sec, Operation Intensity (Arithmetic intensity) for the ten executables
3. [2 points] reasoning about performance change in the ten executables
4. [1 point] your system configuration
5. [1 point] answer to the following question:
    a. A computer has a memory bandwidth of 2 Gwords/byte and a peak performance of 8 Gflops/sec.
    Calculate the ridge point operational intensity.
    Operational intensity of a stencil code is 5 Gflops/word.
    Where does the code fall (memory-region or compute-region)?
    Now, assume that the peak performance is quadrupled (let's say by increasing the number of cores) but the effective memory bandwidth is only doubled. **Find whether the code is memory-bound or compute-bound.**

Installation steps

Please use UVA CS portal for this assignment.

1. Download Intel Advisor from here:
    a. You can choose online installer for your OS and continue download as guest
    b. This will download an ***intel-advisor.*.sh*** file, please source this file in your terminal
        i. You can also find ***intel-advisor-2025.0.0.798.sh*** in ***1_assignment_roofline.tar.gz*** (for linux)
2. Follow the installation instructions on the screen
    a. The installation would be in $HOME/intel/oneapi/advisor or /opt/intel/oneapi/advisor (for root users)
3. Environment variables need to be set
    a. source $HOME/intel/oneapi/setvars.sh
    b. Confirm the following points to your latest installation path
        i. env | grep ADVISOR_
        ii. ADVISOR_2025_DIR=/u/<computing-id>/intel/oneapi/advisor/2025.0
            - For windows: set ADVISOR_2022_DIR="C:\Program Files (x86)\Intel\oneAPI\advisor\latest"
    c. Reference: Setup environment variables (link)

Matrix Multiplication Source code

The source codes and Makefile can be found in ***1_assignment_roofline.tar.xz***

*There are 7 *.cpp files:*

1. 0_vecadd.cpp (Naïve vector addition)
2. 0_gemv.cpp (Naïve matrix-vector multiplication)
3. 0_matmul.cpp (Naïve matrix-matrix multiplication)
4. 1_matmul.cpp (SIMD reduction)
5. 2_matmul.cpp (Transpose one matrix)
6. 3_matmul.cpp (Tiling the matrix multiplication)
    you can modify CHUNK_SIZE and see the impact on the roofline plot
    (put the most performant CHUNK_SIZE value on your assignment report)
7. 4_matmul.cpp (Aligning the vectors)
    has compilation for sse, avx, avx2 ISA

You will have 10 binaries in total.

Makefile:

1. make all builds all the binaries
2. make run compiles the sources and executes all the binaries
3. make clean removes the binaries

you can also build one binary at a time (see the Makefile)

References

Intel Advisor is based on the initial Roofline model from UC Berkeley and improved using Cache-line Aware Roofline Model (CARM) paper.

CARM paper used in Intel Advisor:
Cache-aware Roofline model: Upgrading the loft [*IEEE CAL 2014*]

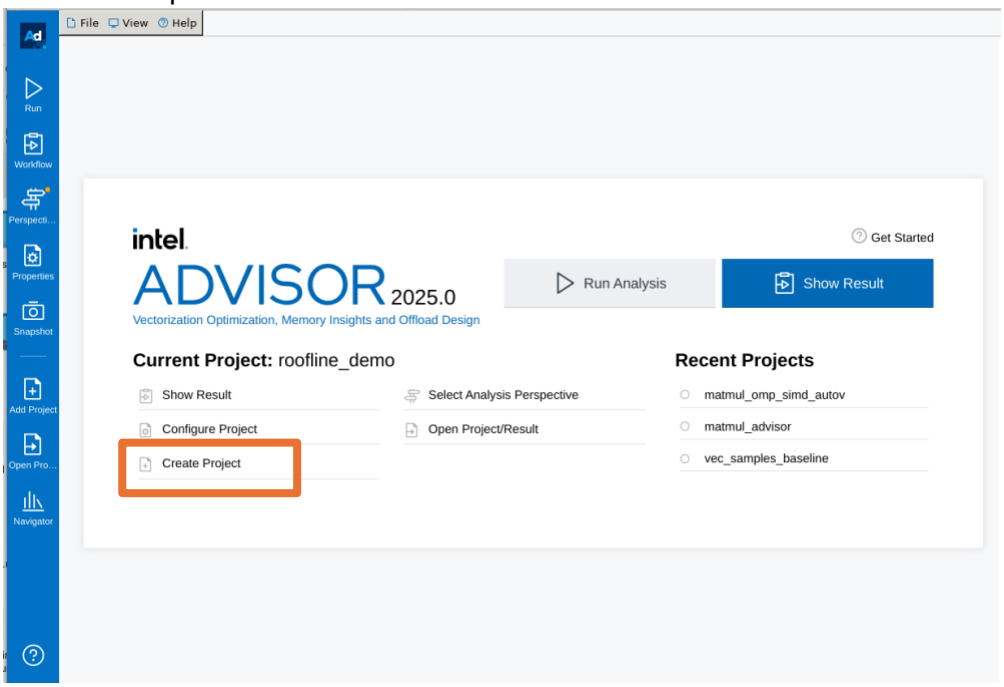
Other papers:

1. Beyond the Roofline: Cache-Aware Power and Energy-Efficiency Modeling for Multi-Cores [*IEEE CAL 2017*]
2. Exploring GPU performance, power and energy-efficiency bounds with Cache-aware Roofline Modeling [ISPASS 2017]
3. Application-driven Cache-Aware Roofline Model [ELSEVIER 2020]

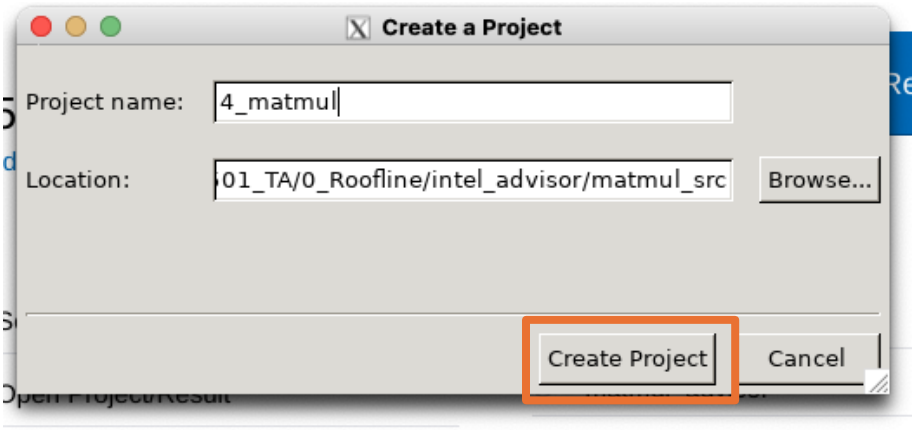
Reference for using Intel Advisor (this one is a little old, but you can get a feel of the tool)
https://www.youtube.com/watch?v=h2QEM1HpFgg

Getting started with Intel Advisor

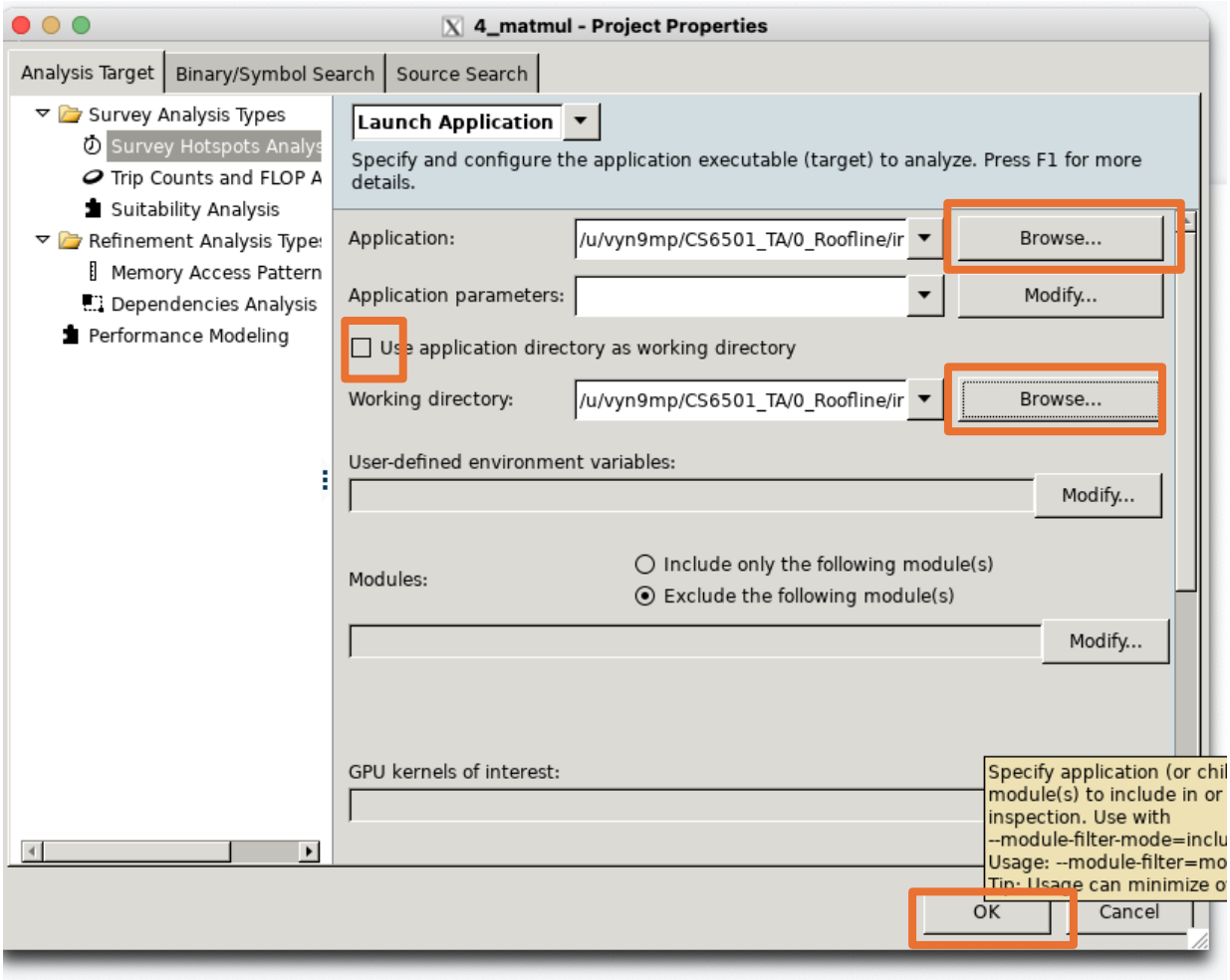
1. Once you have installed Intel Advisor and sources the required setup environment variables (steps here), you can run it with GUI.
2. We recommend using UVA CS portal machines for this assignment. Please login with ssh -X. Disable VPN as the GUI seems to be slow with VPN.
3. [on your terminal] advisor-gui &
    a. This would open a screen as shown below:



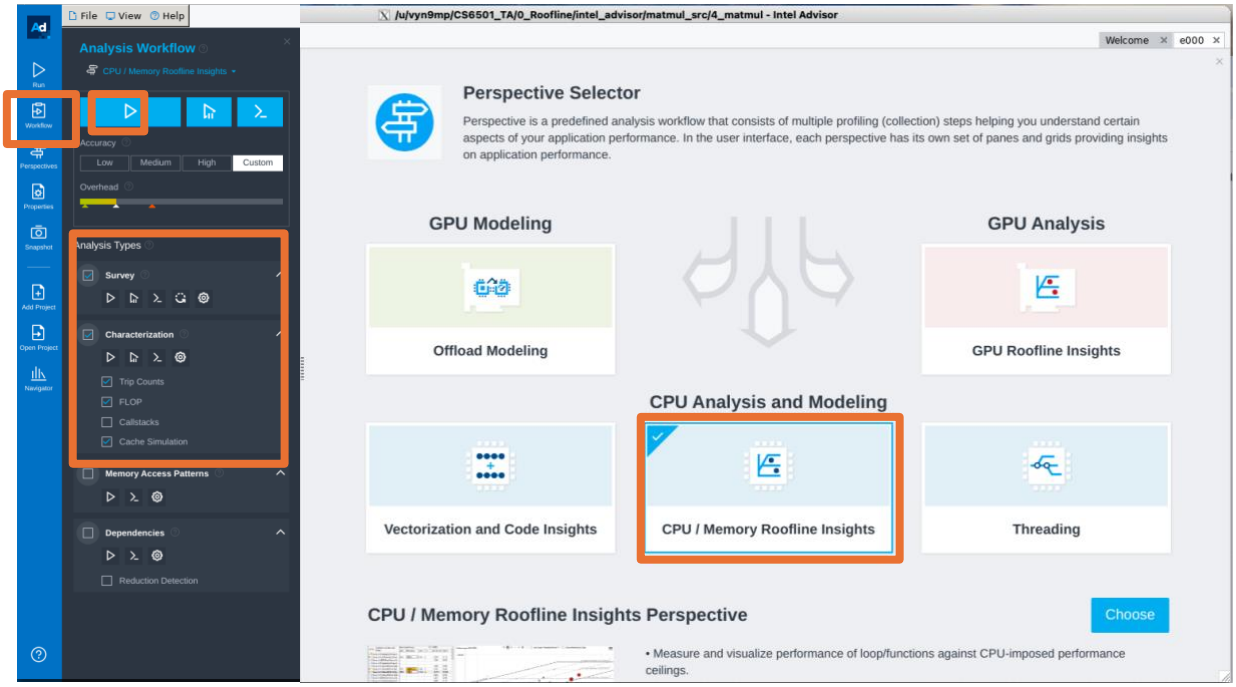
4. Create a Project. Add the location of your project and put a name. (Use different names for the project for each executable)



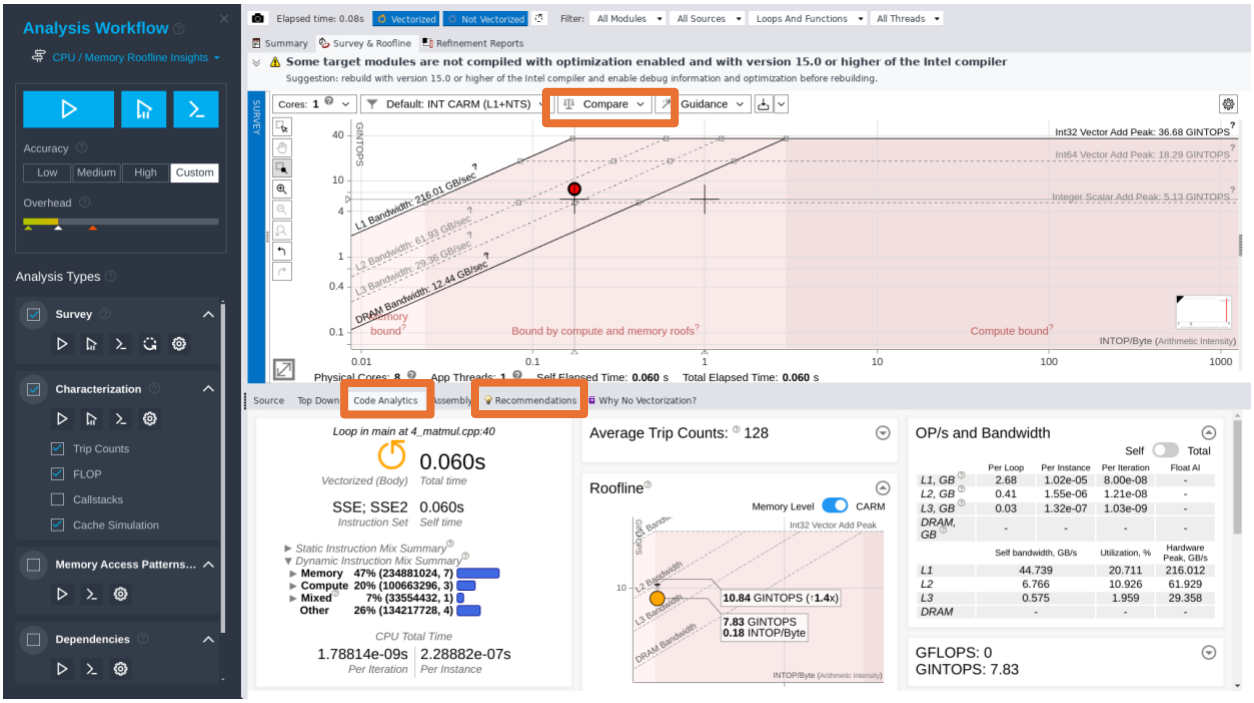
5. This will open a screen as shown below. Select your application (one of the executables you get from assignment_roofline/. Change the working directory to where you created your project. Click OK.



6. Select CPU/Memory Roofline Insights and Workflow from the screen that pops up next. In Analysis Workflow, select Survey, Characterization (Trip Counts, Flops, Cache Simulation)



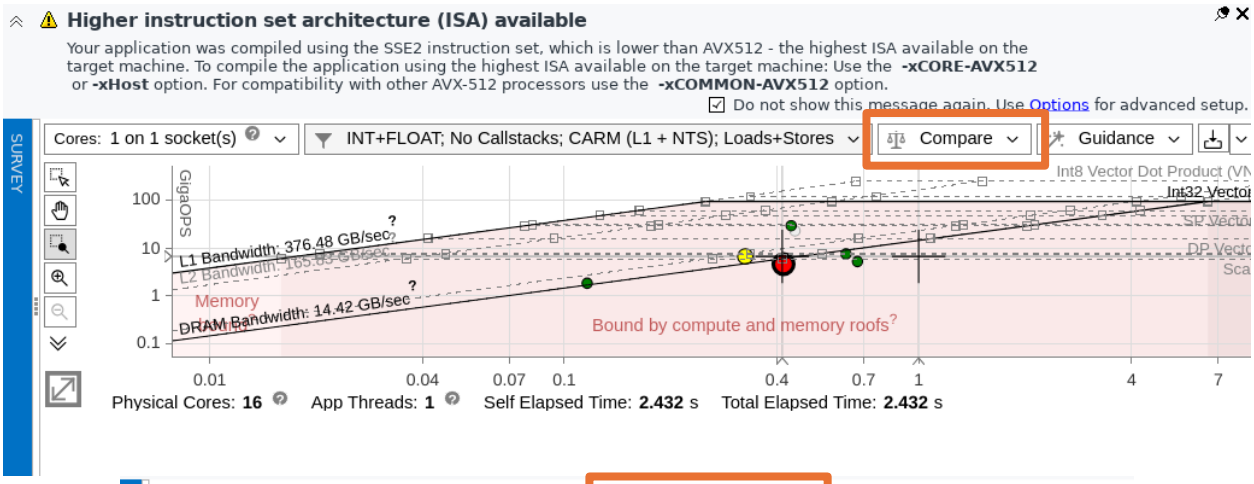
7. Click on the run button. It will take some time.
    a. If you get some error saying application is too small to sample, please add a few iterations in the corresponding source code and recompile and rerun.
8. Once done, you will see something like the following (Survey & Roofline tab)

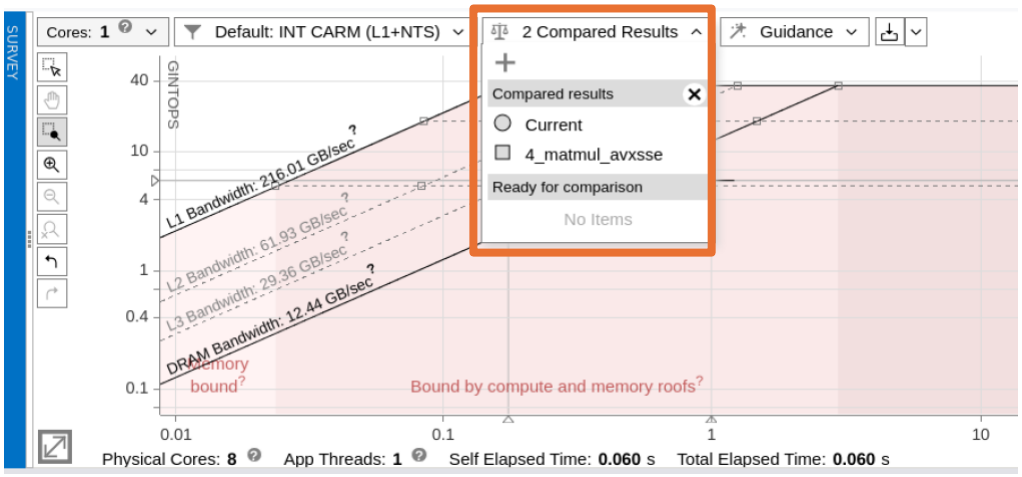


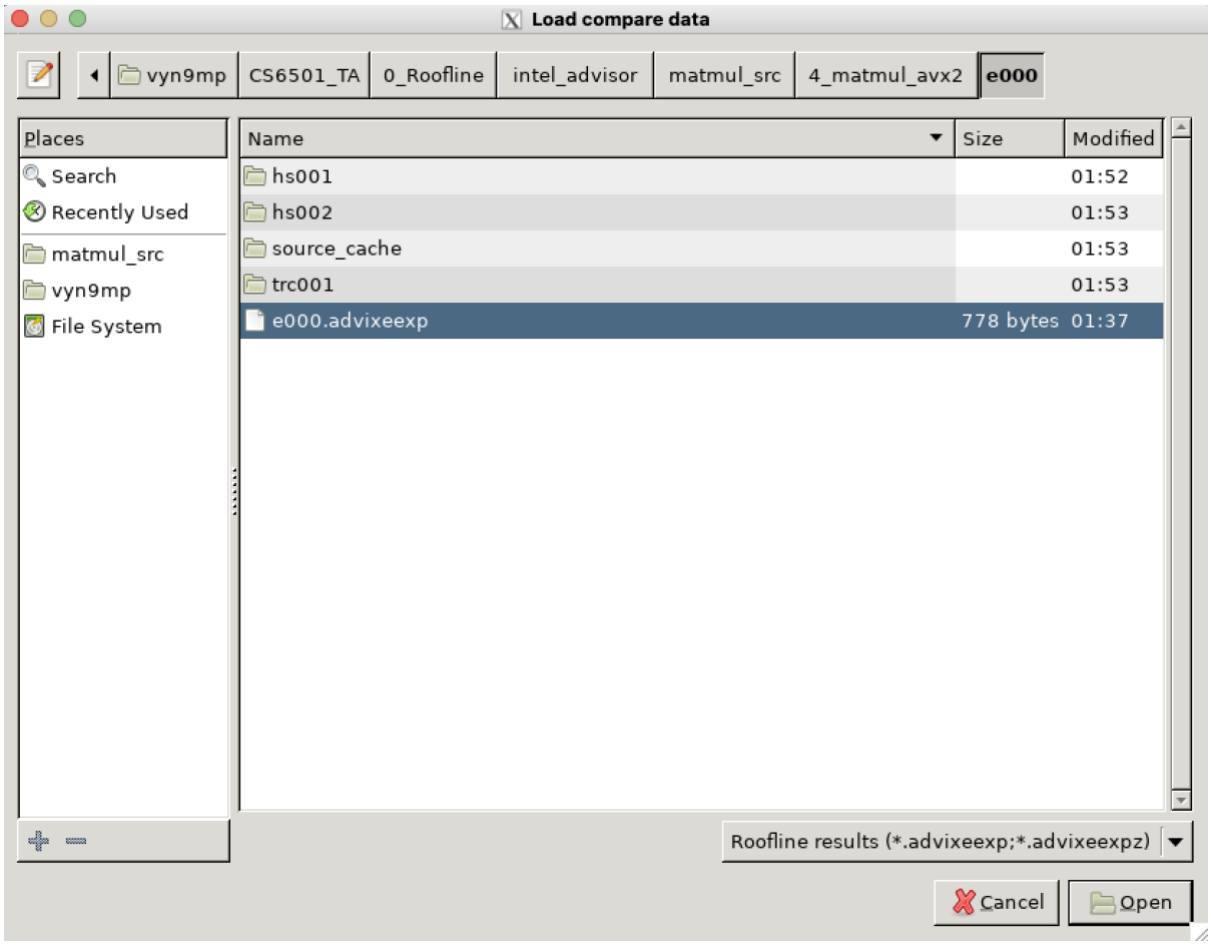
    a. The roofline plot here shows the peak performance of each instruction and peak bandwidth of L1, L2, L3, and DRAM.

b. The red dot shows the position of your application (you also look at the exact line of the code, when you click on it)
c. Code Analytics tab summarizes the analysis
d. Recommendations tab shows possible optimization to improve the performane. Please read through them to understand the bottleneck and how a subsequent executable in assignment_roofline/ solves them.

9. Create new projects for all the 10 executables in assignment_roofline/. Understand the change in position of the red dot. Note down the INTOPS/s and Operational Intensity in a table (for the report)
10. Finally, you add all your projects in one project to compare the roofline plots of each executable within a single plot. Use Compare option as highlighted below. Keep adding projects for all the executables.







11. Once done, you should see something like this. Take a screenshot (zoom in if needed), add to the report (label each of the application marker with the executable)

Roofline plot with all kernels