Roofline Analysis

In 2008, the inventor of RISC-V proposed the concept of roofline analysis in a paper. The examples in the paper focused on AMD Opteron CPUs. Later, the analysis became more popular when NVIDIA implemented it in Nsight, its CUDA GPU profiling toolkit. Since then, many blog posts and YouTube videos have explained roofline analysis. I've been considering how to explain this concept more concisely than most of the resources I've encountered. Here's my attempt.

Roofline Model

The roofline model estimates the upper bound of a chip's performance.

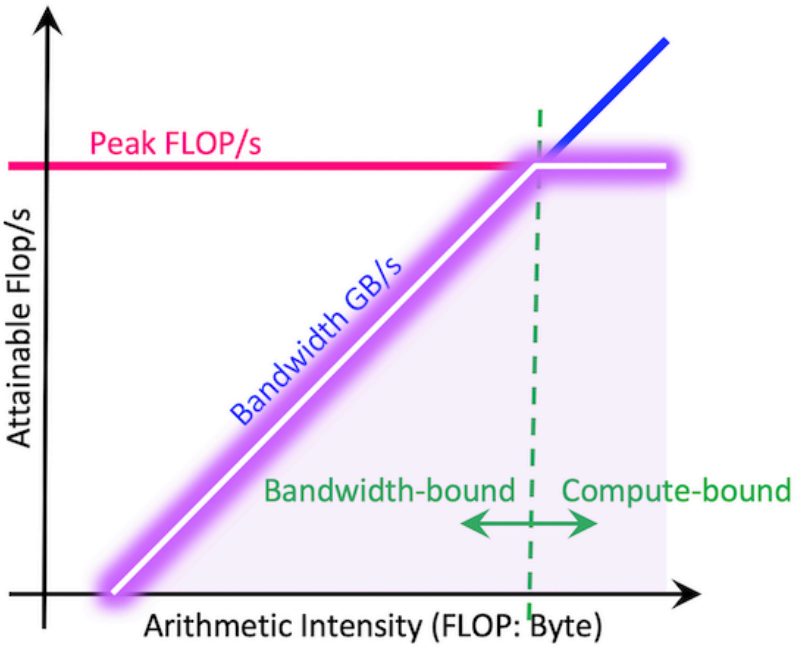
A chip has a peak performance, denoted as π, measured in FLOPS (floating-point operations per second), determined by its computational circuit.

When running an application, the computational circuit often has to wait for data to be loaded or saved. If the memory bandwidth is β (in bytes per second) and the application performs I flops per byte, the maximum achievable performance becomes βI.

Thus, the peak performance is described by a roofline-shaped curve:

$$\max(\beta I, \pi)$$

The term βI depends on I, so we plot this relationship on a 2-dimensional graph, where the x-axis represents *arithmetic intensity* I and the y-axis represents performance in FLOPS.



Log-Log Plot

Modern chips can achieve π in the teraflops range, which is so large that we typically use a log-scale for the y-axis.

The value of I (arithmetic intensity) can range from below 1 to very large values. For example, an element-wise operation on `fp32` tensors must first load a 4-byte `fp32` value before processing it, and then write the 4-byte result. Its arithmetic intensity is thus:

$$\frac{1 \text{ flop}}{8 \text{ bytes}}$$

On the other hand, the multiplication of two `fp16` tensors of size $n \times n$ has an arithmetic intensity of:

$$\frac{2n^3}{2n^2 + 2n^2} = \frac{n}{2}$$

Given that n can reach into the thousands, the arithmetic intensity can also scale into the thousands. This wide range of I suggests that a log-scale is appropriate for the x-axis as well.

Plotting in Log-Log Scale

Plotting π (constant peak performance) on a log-log plot results in a horizontal line, similar to its representation on a linear plot.

However, plotting the linear function $y = \beta I$ on a log-log plot differs from its linear counterpart.

When plotting a point (y, I) in log-log scale, the coordinates are transformed to $(\log y, \log I)$.

The slope of $y = \beta I$ between two points in the range $[I_1, I_2]$ is given by:

$$\frac{\log y_2 - \log y_1}{\log I_2 - \log I_1} = \frac{\log(y_2/y_1)}{\log(I_2/I_1)} = 1$$

Thus, in a log-log plot, all linear functions have a slope of 45 degrees.

The intercept occurs when the x-coordinate $\log I = 0$, or $I = 1$, giving us:

$$\log(\beta \cdot 1) = \log \beta$$

Therefore, the value of β, which defines the slope in a linear plot, determines the intercept in a log-log plot.

When $\beta < 1$, $\log \beta < 0$. When $\beta > 1$, $\log \beta > 0$.

Performance Tuning

Use Lower-bits

Inference in deep neural networks using `fp16` typically does not lead to a significant loss in precision compared to `fp32`. However, using `fp16` halves the number of bytes that need to be loaded and saved, effectively doubling the arithmetic intensity I.

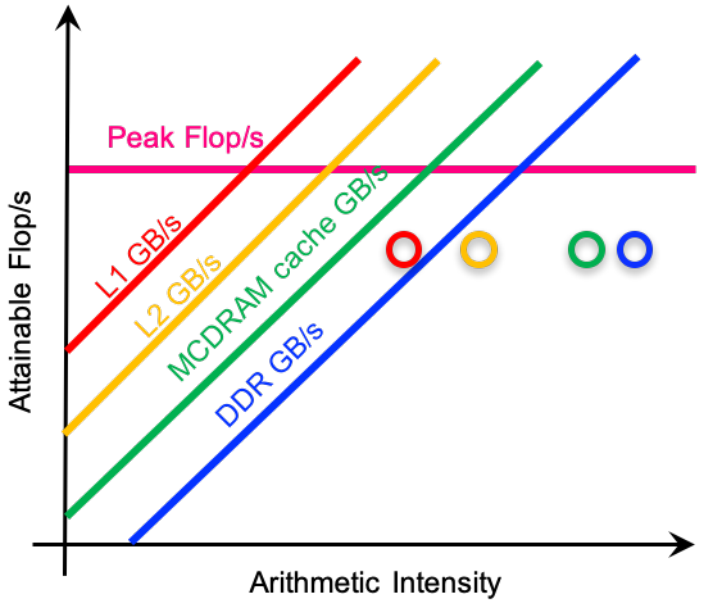
Suppose that, prior to this doubling, I lies below the diagonal line on the roofline model, meaning the application is constrained by memory bandwidth—i.e., it is memory-bound. By doubling I to $2I$, may make this new value lies below the horizontal line representing peak performance, the application can potentially shift from being memory-bound to achieving the chip's peak performance.

Operation Fusing

Another commonly used optimization is **operation fusion**, which reduces the number of loads and saves for intermediate results, often referred to as activations. This optimization also increases the arithmetic intensity I, helping the application get closer to the peak performance.

Suboptimal Memory Utilization

In some cases, an application may fail to fully utilize the available memory bandwidth β, resulting in an effective bandwidth β' that is less than β. Since the bandwidth defines the intercept in the log-log plot, the diagonal line corresponding to $\beta' I$ would have the same 45-degree slope but a lower intercept compared to βI.



Suboptimal Computational Utilization

Similarly, a lack of certain optimization skills may prevent an application from fully utilizing the chip's peak performance π. In this case, the actual performance would be represented by a lower value, π', which would appear as a horizontal line below the peak π in the roofline plot.