Michigan State University

**CMSE/CSE 822 – Parallel Computing** **Fall 2019**

**Homework 2**

Alexander Harnisch

---

# 1) Performance Modeling

## a)

There is some missing information here and assumptions have to be made. However, it's not important since it only effects the numbers to plug in and get out, not the logic behind it.

The kernel performs 6 floating point operations (FLOP) in each iteration. Assuming no caching we have seven floating point number reading and one writing operation for each loop iterations. Assuming single precision with 4 byte per float that translates to 32 byte of memory acces and an arithmetic intensity $I$ of

$$I = \frac{6\,\text{FLOP}}{32\,\text{byte}} = \frac{3}{16}\,\frac{\text{FLOP}}{\text{byte}}\,. \tag{1}$$

For double precision we obviously get half of that. Assuming some caching where at least the same variable has to be loaded only once per loop iteration, we would have just 2 reading operations and therefore 8 byte of memory access which would result in $I = 3/4\,\text{FLOP/byte}$ for single and half that for double precision. Depending on many factors (caching, compiler used, etc.) the actual numbers might be different.

## b)

In a simple roofline model for some $I$ the critical peak performance $\pi_{\text{crit}}$ is given by $\beta I$, where $\beta$ is the peak memmory bandwith. So whatever the assumptions and resulting $I$ from a) is, for a peak performance of larger than $\pi_{\text{crit}}$ the kernel would be compute bound, otherwise memory bound.

Let's say we have double precision and some caching, so $I = \frac{3}{8}\,\frac{\text{FLOP}}{\text{byte}}$ we would get

$$\pi_{\text{crit}} = 30\,\frac{\text{GB}}{\text{s}} \cdot \frac{3}{8}\,\frac{\text{FLOP}}{\text{byte}} = 11.25\,\frac{\text{GFLOP}}{\text{s}}\,. \tag{2}$$

**c)**

A simple roofline model plot is given by Figure 1. The performance for an arithmetic intensity of $I = 3/8\,\mathrm{FLOP/byte}$ is $11.25\,\mathrm{GFLOP/s}$.

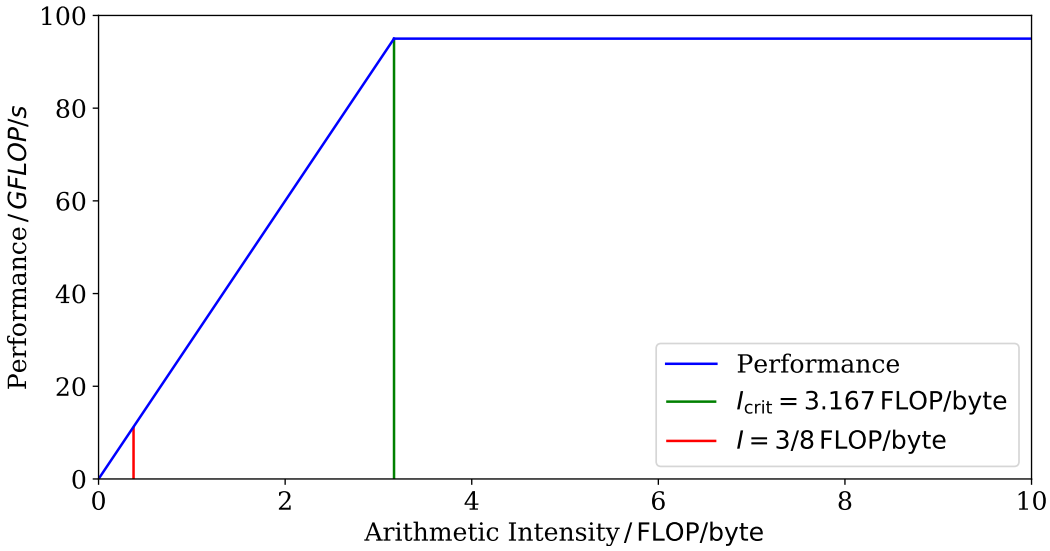


**Figure 1:** Simple roofline model.

## 2) Cache optimization: Matrix Vector Multiplication