Project 1: Memory Hierarchies

Team 7: Ian Freeman, Nick Schwartz, Anna Brandl, Jingyao Tang, Xinrui Yu

Warm-Up

Kernel	Arithmetic Intensity (FLOPs/byte)
Y[j] += Y[j] + A[j][i] * B[i]	3/32
s += A[i] * A[i]	1/4
s += A[i] * B[i]	1/8
Y[j] += A[j] + C * B[i]	1/12

Part 1: Matrix – Matrix Multiplication

1. Code is uploaded to the team GitHub: https://github.com/cmse822/project-1-seven-c-s

2. For a given matrix size N, the total number of floating-point operations performed by this operator is:

$$nn(2n-1) \;=\; 2n^3 - n^2$$

3. Performance in Mflop/s of the matrix-matrix multiplication for N = 100 *only tested using one core
   - Architecture #1 HPCC Intel 18: 1685.78 Mflop/s
   - Architecture #2 HPCC Intel 16: 1852.14 Mflop/s

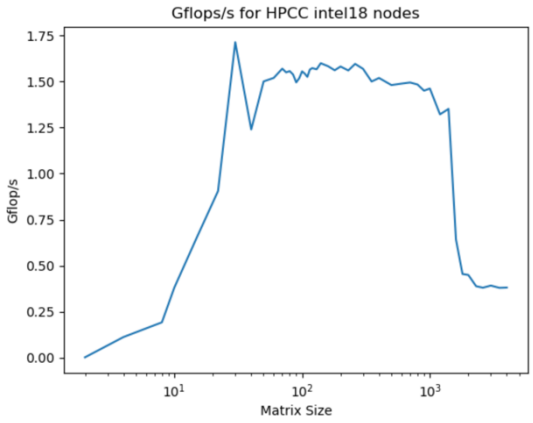
4. Theoretical Peak Performance:

Architecture	Clock Speed	Cache Size / Layout	Theoretical Peak Performance	Comparison
HPCC Intel 18	2.40 GHz	L1d cache: 32K L1i cache: 32K L2 cache: 1024K L3 cache: 28160K	20 Cores 2.40 GHz TPP: 48 Gflop/s for the entire system (all cores) One core: 2.4 Gflop/s	The theoretical performance is higher than the performance calculated in number 3. When comparing one core to one core, the performances are similar. For example: HPCC Intel 18 has a theoretical peak performance for one core of 2.4 Gflop/s and the actual performance was 1.685 Gflop/s.
HPCC Intel 16	2.40 GHz	L1d cache: 32K L1i cache: 32K L2 cache: 256K L3 cache: 35840K	14 Cores 2.40 GHz TPP: 33.6 Gflop/s for the entire system (all cores) One core: 2.4 Gflop/s	

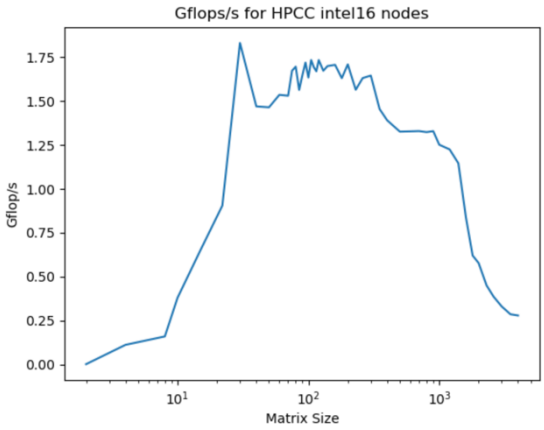
Talking with the TA: Theoretical Peak Performance = Clock Speed * # of Cores * 1 (given)

5. Plots Measuring Gflop/s vs. N:

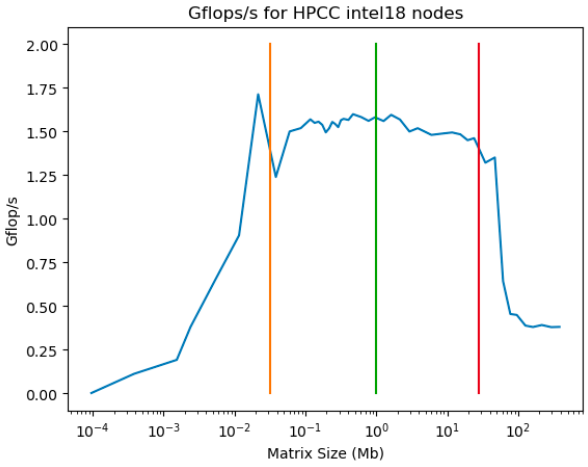
HPCC Intel 18

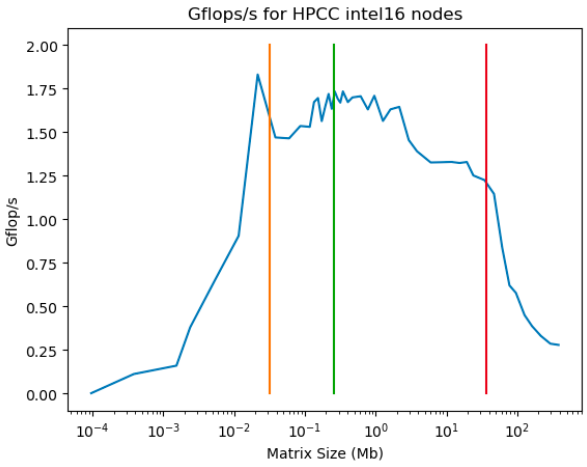


HPCC Intel 16



6. Explanation of measured performance:





Our graphs above show drops at the orange, green and red lines that represent the cache level changes. The orange lines are L1, green lines are L2, and red lines are L3.

These occur at different spots depending on the cache sizes of each of the architectures.

The cache sizes in megabytes were calculated using the approximation $\frac{8n^2}{10^6} \cdot 3$ (8 bytes per float, n^2 floats per matrix, 3 matrices per calculation).

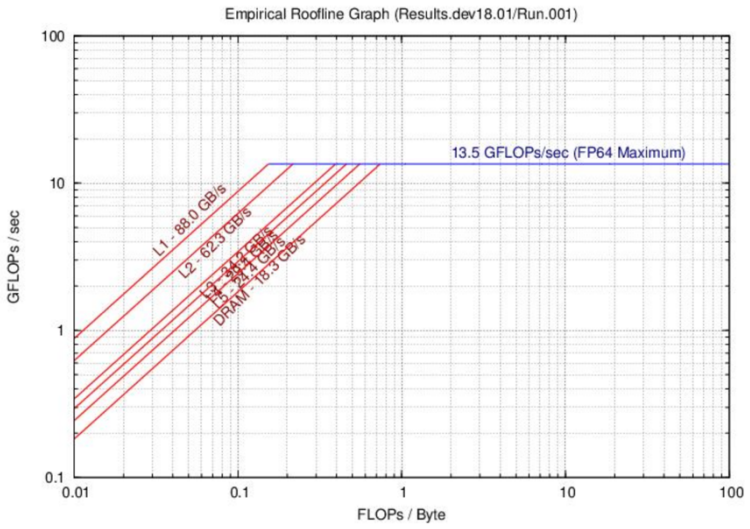
Part 2: The Roofline Model

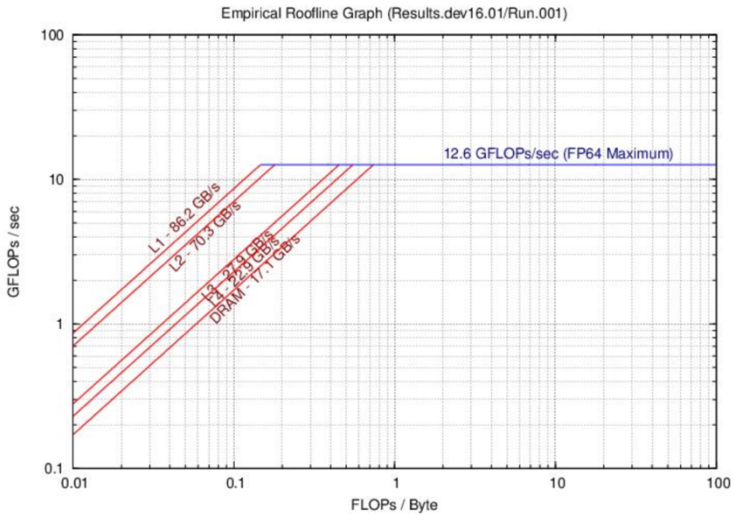
1. Reviewing reference materials.
2. Cloning CS Roofline Toolkit & modify the Config file for each machine's architecture.

3. Running ERT in serial mode & reporting bandwidth and peak performances for the following:

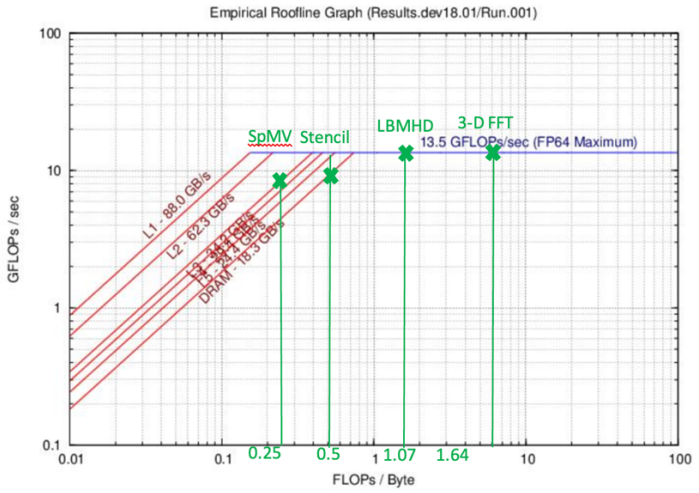
Architecture	Metric	L1	L2	L3	DRAM	Ridge Point
HPCC Intel 18	Bandwidth	88.0 GB/s	62.3 GB/s	34.2 GB/s	18.3 GB/s	Always where the diagonal red lines meet the horizonal blue line. L1: ~ 0.15 L2: ~ 0.22 L3: ~ 0.4 DRAM: ~ 0.7
	Peak Performance	13.2 Gflop/s	13.5 Gflop/s	13.5 Gflop/s	12.81 Gflop/s	
HPCC Intel 16	Bandwidth	86.2 GB/s	70.3 GB/s	27.9 GB/s	17.1 GB/s	Always where the diagonal red lines meet the horizonal blue line. L1: ~ 0.15 L2: ~ 0.18 L3: ~ 0.45 DRAM: ~ 0.7
	Peak Performance	12.6 Gflop/s	12.6 Gflop/s	12.55 Gflop/s	11.97 Gflop/s	

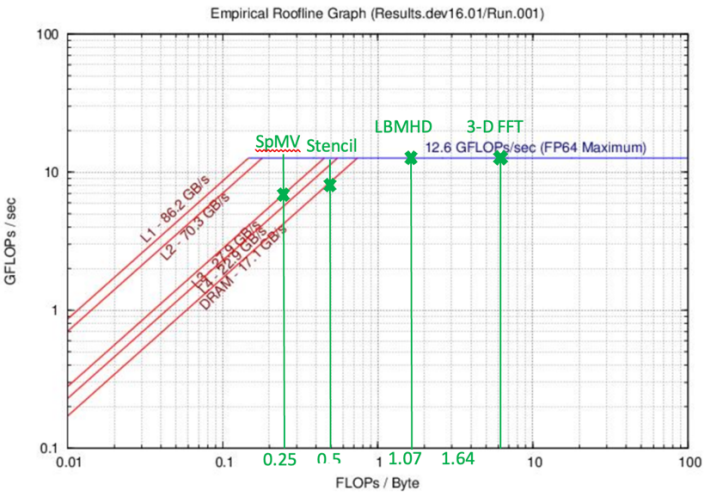
Talking with the TA: Performance Calculation: https://crd.lbl.gov/assets/pubs_presos/parlab08-roofline-talk.pdf // Slide 12





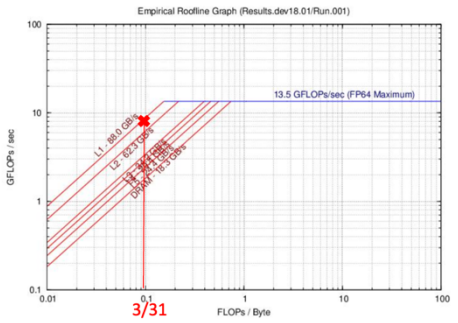
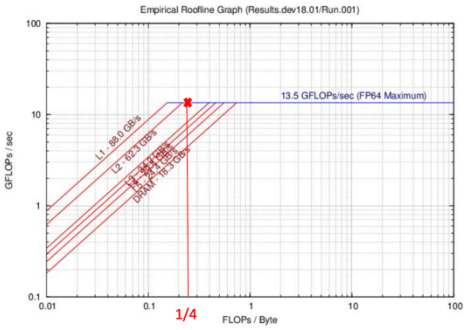
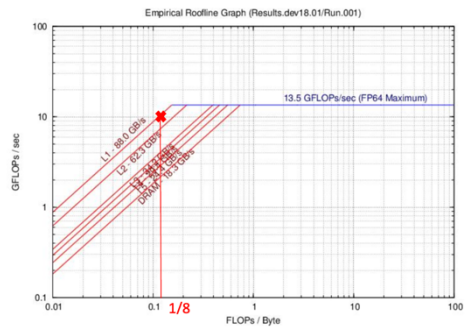
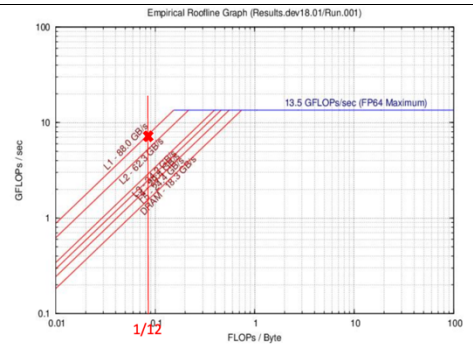
4. Considering the Four FP kernels in “Roofline: An Insightful Visual Performance Model for Floating-Point Programs and Multicore Architectures”:

<table>
<tr><th>Kernel</th><th>Performance on Platforms</th><th>Optimization Strategy</th><th>Graphs</th></tr>
<tr><td rowspan="2">SpMV</td><td>HPCC Intel 18: Limited by the L3 bandwidth</td><td rowspan="2">Have the code only utilize the L1 and L2 cache size to minimize the amount of cache misses.</td><td rowspan="8"> 

Four kernels perform on the HPCC Intel 18


Four kernels perform on the HPCC Intel 16</td></tr>
<tr><td>HPCC Intel 16: Limited by the L3 bandwidth</td></tr>
<tr><td rowspan="2">LBMHD</td><td>HPCC Intel 18: Limited by the CPU</td><td rowspan="2">Use more cores if possible.</td></tr>
<tr><td>HPCC Intel 16: Limited by the CPU</td></tr>
<tr><td rowspan="2">Stencil</td><td>HPCC Intel 18: Limited by the DRAM</td><td rowspan="2">Avoid using the DRAM and stay within the caches.</td></tr>
<tr><td>HPCC Intel 16: Limited by the DRAM</td></tr>
<tr><td rowspan="2">3D - FFT</td><td>HPCC Intel 18: Limited by the CPU</td><td rowspan="2">Use more cores if possible.</td></tr>
<tr><td>HPCC Intel 16: Limited by the CPU</td></tr>
</table>

5. Considering the four kernels given in the warm-up above:

Kernel	Performance on Platforms	Optimization Strategy	Graphs
Y[j] += Y[j] + A[j][i] * B[i]	HPCC Intel 18: Limited by the L1 bandwidth. HPCC Intel 16: Limited by the L1 bandwidth.	Design the code so that it utilizes the L1 cache as much as possible.	
s += A[i] * A[i]	HPCC Intel 18: Limited by the L3 bandwidth. HPCC Intel 16: Limited by the L3 bandwidth.	Have the code only utilize the L1 and L2 cache size to minimize the amount of cache misses.	
s += A[i] * B[i]	HPCC Intel 18: Limited by the L1 bandwidth. HPCC Intel 16: Limited by the L1 bandwidth.	Design the code so that it utilizes the L1 cache as much as possible.	
Y[j] += A[j] + C * B[i]	HPCC Intel 18: Limited by the L1 bandwidth. HPCC Intel 16: Limited by the L1 bandwidth.	Design the code so that it utilizes the L1 cache as much as possible.	

6. Comparison of results from roofline model to the matrix-matrix multiplication operation:

The roofline model found our ideal peak performance to be 12.6 Gflop/s for Intel 16 and 13.5 Gflop/s for Intel 18. However, we found our peak performance to be 1.685 Gflop/s for Intel 18 and 1.852 Gflop/s for Intel 16. This discrepancy likely stems from our matrix multiplication code, which was not maximally optimized or parallelized. Additionally, the roofline model finds the upper bound on performance by using the peak bandwidth and peak performance. Our performance is lower than the roofline's peak because ours is limited by bandwidth. We found consistent performance drops after each cache filled, and after our matrices reach L3 cache size, we are limited by DRAM. These dips are explained by peak bandwidth limitations from each memory architecture.