Nicholas Skinner

CS475

Spring 2019

**Project #4 - Vectorized Array Multiplication and Reduction using SSE**

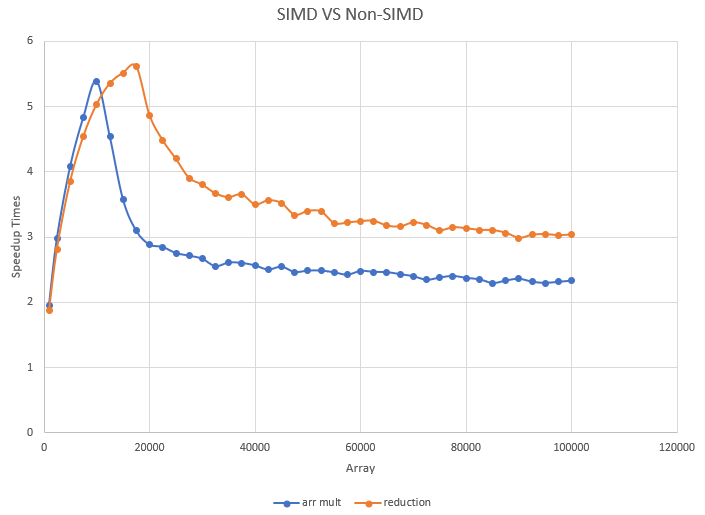
**Environment:**

I ran my program on the Oregon State University FLIP server.

**Table:**

|  |  |  |
| --- | --- | --- |
|  | arr mult | reduction |
| 1000 | 1.957 | 1.879 |
| 2500 | 2.984 | 2.817 |
| 5000 | 4.08 | 3.85 |
| 7500 | 4.836 | 4.543 |
| 10000 | 5.388 | 5.032 |
| 12500 | 4.542 | 5.352 |
| 15000 | 3.577 | 5.514 |
| 17500 | 3.096 | 5.615 |
| 20000 | 2.885 | 4.872 |
| 22500 | 2.847 | 4.483 |
| 25000 | 2.754 | 4.198 |
| 27500 | 2.714 | 3.905 |
| 30000 | 2.666 | 3.802 |
| 32500 | 2.552 | 3.667 |
| 35000 | 2.608 | 3.609 |
| 37500 | 2.602 | 3.653 |
| 40000 | 2.564 | 3.497 |
| 42500 | 2.5 | 3.561 |
| 45000 | 2.545 | 3.518 |
| 47500 | 2.463 | 3.339 |
| 50000 | 2.485 | 3.398 |
| 52500 | 2.485 | 3.389 |
| 55000 | 2.458 | 3.21 |
| 57500 | 2.422 | 3.22 |
| 60000 | 2.48 | 3.242 |
| 62500 | 2.465 | 3.246 |
| 65000 | 2.461 | 3.172 |
| 67500 | 2.429 | 3.161 |
| 70000 | 2.398 | 3.223 |
| 72500 | 2.344 | 3.183 |
| 75000 | 2.379 | 3.103 |
| 77500 | 2.402 | 3.148 |
| 80000 | 2.372 | 3.132 |
| 82500 | 2.351 | 3.107 |
| 85000 | 2.293 | 3.104 |
| 87500 | 2.333 | 3.063 |
| 90000 | 2.359 | 2.986 |
| 92500 | 2.313 | 3.034 |
| 95000 | 2.296 | 3.043 |
| 97500 | 2.314 | 3.026 |
| 100000 | 2.33 | 3.039 |

**Graph:**

****

**Patterns commentary:**

* What are we seeing?

I believe I am seeing the SIMD have a slow start, and as it progresses it improves its performance. After it reaches its peak, it suffers from a form of performance decay as more and more arrays are added to the equation.

Overall, it looks like the SIMD version is an average of ~2.75 times for the array multiplication, and an average of ~3.5 times for the reduction version.

* Is it consistent? why/why not?

It is not consistent, we appear to have a massive amount of buildup in performance in the beginning of the chart, due to fewer arraymults. After some time, we appear to still get some performance decay, but it appears to mostly level out.

The variance by the reduction is easier to analyze, we seem to go from a just under 2 times speedup, up to an approximate 5.5 times speedup, then decay down to a 3 times speedup, still great, but not as impressive as its peak.

* Knowing that SSE SIMD is 4-floats-at-a-time, why could you get a speed-up of < 4.0 or > 4.0 in the array-multiplication?

The variance on the machine itself could be a big factor, it appears to have been one in the previous assignments, causing some issues on analyzing some of these results accurately.

I imagine SIMD is needing to load and store too many operations, and it is not doing the CPU calculations effectively.

In addition to the innate machine variance, I imagine implementing concepts such as prefetching could attain a more consistent performance.

* Knowing that SSE SIMD is 4-floats-at-a-time, why could you get a speed-up of < 4.0 or > 4.0 in the array-multiplication-reduction?

Again, machine variance could be a big factor in play here, if we get a lowball run for our non-SIMD, and a peak performance from our SIMD, it can cause some excessively large numbers in the performance.

I imagine SIMD is needing to load and store too many operations, and it is not doing the CPU calculations effectively.

In addition to the innate machine variance, I imagine implementing concepts such as prefetching could attain a more consistent performance.