Name: Athit Vue

CSCI 551

Program 1: Vectorization Report

**Objectives:**

* Predict speedup due to vectorization and why.
* Record actual times on three runs each for mmvec and mmnovec using n=8192.
* Record maximum resident set size.
* Discuss how you approached tuning and how successful you think it was.

**Data Collected:**

**mmnovec**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Run# | User time (secs) | System time(secs) | Total(secs) | Max Resident Set Size |
| 1 | 271.36 | 0.06 | 271.42 | 7.88E+05 |
| 2 | 270.86 | 0.08 | 270.94 | 7.88E+05 |
| 3 | 273.01 | 0.06 | 273.07 | 7.88E+05 |

**mmvec**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Run# | User time (secs) | System time(secs) | Total(secs) | Max Resident Set Size | Speedup |
| 1 | 248.71 | 0.05 | 248.76 | 7.88E+05 | 1.091 |
| 2 | 248.75 | 0.06 | 248.81 | 7.88E+05 | 1.089 |
| 3 | 248.56 | 0.08 | 248.64 | 7.88E+05 | 1.098 |

**Prediction:**

I thought that I could at least get a speedup of at least 1.2 due to using data alignment and creating dynamically allocated arrays, since I know that with my code some of the for-loops have data dependencies that will not vectorized. However, I lost almost 0.2 of speedup due to my unsuccessful attempt at data aligning. Some of my code had data dependencies that I couldn’t somehow get rid of so I knew my speedup time was not going to be that great from non-vectorization.

**Methods:**

I tried to implement the multiplication of the two matrices following the guidelines as provided by: <https://software.intel.com/en-us/node/684329> and by using brute force. I tried to avoid many things the guideline says to avoid. I tried to avoid: mixing data-types in the same loop, data-dependent loop exit conditions, loop iteration dependencies, and pointers. However, some of these things were unavoidable and according to the optimization report generated, I am penalized for it. Some of the things that I try to do to achieve a better run time was: using one statement inside nested loops, use unit strides withing the foremost nested loop, use malloc to create dynamic arrays, use the keyword restrict to tell the compiler that pointers have no alias, use pragma to let the compiler know to ignore some stuff or take some stuff into consideration, try to aligned memory using mm\_malloc and aligned\_malloc, and correct some of the stuff the optimization report was hinting to me. However, I was unable to achieve correcting anything the optimization report hinted before the deadline.

**Conclusion:**

A lot of these things were new to me and I wished I had more time to work on this as it has been frustrating and interesting at the same time on trying to find the solution for achieving better auto-vectorization. The tutorial in class was helpful, however, I couldn’t find much examples elsewhere.

In conclusion, I could have vectorized this more fully if I was able to successfully implement data alignment of the arrays.