Patience Lamb

CS 3337

Pt 1.1-3

Using the lowest GCC optimization settings of compiling with the flag of -O0 and averaging the time it takes to do a column and row transversal, the resutls are as follows:

***PT 1 Comparing averages over 5 times with low optimization and rows first***

*The average time it takes for a 100 mapped array to run 5 times is 2.96e-05*

*The average time it takes for a 100 array of pointers to run 5 times is 2.22e-05*

*The average time it takes for an 100 auto array to run 5 times is 2.62e-05*

*The average time it takes for a 1000 mapped array to run 5 times is 0.0025614*

*The average time it takes for a 1000 array of pointers to run 5 times is 0.0017788*

*The average time it takes for an 1000 auto array to run 5 times is 0.002122*

*The average time it takes for a 10000 mapped array to run 5 times is 0.281855*

*The average time it takes for a 10000 array of pointers to run 5 times is 0.186534*

*The average time it takes for an 10000 auto array to run 5 times is 0.23763*

***PT 1 Comparing averages over 5 times with low optimization and columns first***

*The average time it takes for a 100 mapped array to run 5 times is 1.8e-05*

*The average time it takes for a 100 array of pointers to run 5 times is 8.84625*

*The average time it takes for an 100 auto array to run 5 times is 1.5e-05*

*The average time it takes for a 1000 mapped array to run 5 times is 0.002035*

*The average time it takes for a 1000 array of pointers to run 5 times is 8.88612*

*The average time it takes for an 1000 auto array to run 5 times is 0.0016436*

*The average time it takes for a 10000 mapped array to run 5 times is 0.705397*

*The average time it takes for a 10000 array of pointers to run 5 times is 17.3072*

*The average time it takes for an 10000 auto array to run 5 times is 0.691631*

1. The column major initially has a lesser runtime than the column major but with a larger array, the row major has a lesser runtime than the column major.

2. The runtime of the mapped array is an order of magnitude or two lower than the runtime of the array of pointers.

3. The access times of the “auto array” is significantly faster than the array of pointers and marginally faster than the mapped array.

Pt 1.4-8

Using the highest GCC optimization settings of compiling with the flag of -O3 and averaging the time it takes to do a column and row transversal, the resutls are as follows:

***PT 1 Comparing averages over 5 times with high optimization and rows first***

*The average time it takes for a 100 mapped array to run 5 times is 1.4e-05*

*The average time it takes for a 100 array of pointers to run 5 times is 3.2e-06*

*The average time it takes for an 100 auto array to run 5 times is 4e-06*

*The average time it takes for a 1000 mapped array to run 5 times is 0.0003344*

*The average time it takes for a 1000 array of pointers to run 5 times is 0.0002492*

*The average time it takes for an 1000 auto array to run 5 times is 0.0003228*

*The average time it takes for a 10000 mapped array to run 5 times is 0.189103*

*The average time it takes for a 10000 array of pointers to run 5 times is 0.163822*

*The average time it takes for an 10000 auto array to run 5 times is 0.0777452*

***PT 1 Comparing averages over 5 times with high optimization and columns first***

*The average time it takes for a 100 mapped array to run 5 times is 6e-06*

*The average time it takes for a 100 array of pointers to run 5 times is 4.45473*

*The average time it takes for an 100 auto array to run 5 times is 3e-06*

*The average time it takes for a 1000 mapped array to run 5 times is 0.0005554*

*The average time it takes for a 1000 array of pointers to run 5 times is 4.4648*

*The average time it takes for an 1000 auto array to run 5 times is 0.0004346*

*The average time it takes for a 10000 mapped array to run 5 times is 0.673331*

*The average time it takes for a 10000 array of pointers to run 5 times is 10.4736*

*The average time it takes for an 10000 auto array to run 5 times is 0.352942*

4. The highest optimization setting reduced the runtime by a noticeable amount, sometimes a magnitude lower and sometimes a decimal or two different, but eitherway the runtime was decreased.

5. The reason row major is so much faster at larger sizes is because memory is stored in the cache in a row major order so the memory addresses can be accessed in a contiguous order instead of bouncing around.

6. An auto array is faster than a mapped or pointer array due to the fact that it determines the most optimal access at runtime, way more optimal than a human could write an array access.

7. The extra loop prior to timing ensures the memory addresses are loaded prior to access, speeding up the access time.

8. The array of pointers is great for pushing different values into an array whereas the mapped array is good for all one type of values in an array with fast access. The auto array is the fastest but only really practical when you have an array of a specified, constant size.

PT2

1. Using the lowest optimization settings, the results of using rand, randFoo, and default\_random\_engine are as follows:

***PT 2 Comparing runtime of rand, default\_random\_engine, and randFoo (lowest optimization)***

*1.89899e-06 is the average time it takes for rand() to run 99 times.*

*1.0202e-06 is the average time it takes for randFoo() to run 99 times.*

*1.06061e-06 is the average time it takes for default\_random\_engine to run 99 times.*

Using the highest optimization settings, the results of using rand, randFoo, and default\_random\_engine are as follows:

***PT 2 Comparing runtime of rand, default\_random\_engine, and randFoo (highest optimization)***

*1.92929e-06 is the average time it takes for rand() to run 99 times.*

*6.36364e-07 is the average time it takes for randFoo() to run 99 times.*

*6.46465e-07 is the average time it takes for default\_random\_engine to run 99 times.*

2. The highest optimization setting of -O3 produced faster random numbers by an order of magnitude. The randFoo() overall had the fastest number generation marginally below the default\_random\_engine’s runtime.