The method used to compute repeated 2x2 matrix multiply on CUDA involves giving each thread a certain number of matrices C to compute the partial multiplication of , have a thread multiply all the partial results in the block of size B threads, for G blocks to come to an answer. In the coalesced version the threads read the values they need to have for all shared matrices in the block in in a staggered manner for each thread, rather than a thread having a continuous block. In all my tests that follow the coalesced version is faster than the non coalesced version by some speedup.

I played with B, C, and G trying to hone in on a decent machine optimum.

Some data about different input sizes:

```
B: 8 C: 240 G: 100000 MM2chain00 gpu time: 0.09597125 MM2chain00 gpu time: 0.077036
speedup: 1.2457974193883379
B: 16 C: 120 G: 100000 MM2chain00 gpu time: 0.07833 MM2chain00 gpu time: 0.047215
speedup: 1.659006671608599
B: 32 C: 60 G: 100000 MM2chain00 gpu time: 0.0679 MM2chain00 gpu time: 0.026
speedup: 2.5197200722992075
B: 60 C: 32 G: 100000 MM2chain00 gpu time: 0.05658875 MM2chain00 gpu time: 0.024579
speedup: 2.30232108710688
B: 30 C: 32 G: 200000 MM2chain00 gpu time: 0.0563137 MM2chain00 gpu time: 0.023
speedup: 2.4255394753844164
B: 60 C: 16 G: 200000 MM2chain00 gpu time: 0.05060025 MM2chain00 gpu time: 0.0205
speedup: 2.465959209532396
B: 30 C: 16 G: 400000 MM2chain00 gpu time: 0.0509205 MM2chain00 gpu time: 0.020
speedup: 2.538884387659707
B: 60 C: 8 G: 400000 MM2chain00 gpu time: 0.05028775 MM2chain00 gpu time: 0.01896
speedup: 2.652237546478205
B: 120 C: 4 G: 400000 MM2chain00 gpu time: 0.03339425 MM2chain00 gpu time: 0.0186
speedup: 1.79
B: 60 C: 4 G: 800000 MM2chain00 gpu time: 0.0341 MM2chain00 gpu time: 0.0188
speedup: 1.818
```

All of these experiments were run 4 times and the means were taken for the times, at from the 3 times given by the program, only the time on the Time to compute[GPU] was used in the calculations of these values.

It seems that maximum speed seems to come from cases where the number of matrices each thread computes is low. Maybe this is because each thread has to deal with less memory. The maximum speedup was with 120 threads and only 4 matrices per thread, and increasing the number of blocks to 4 times bigger. The difference is quite significant between different values of B C and G, with the worst for coalesced being .077 to .0186 at the best, a 4x speedup. I’m not really sure why the sizes that work the best work well, it might be due to the specific graphic card hardware structure.