Trang Do

April 27, 2020

Advanced Cryptology Final

Due to time constraints, my original plan to do a comparison between a brute force factorization, Quadratic Sieve factorization, and parallel Quadratic Sieve factorization of semiprimes did not pan out. Instead what I did was parallelize Pollard Rho and compared it to two variations of serial Pollard Rho factorization. All code references had been detailed in the source code, “kernel.cu”.

Using a list of primes I found ranging from 593,441,861 to 613,651,349, I created an array of 50 elements with the semiprimes of these prime values. The semiprimes, ‘N’, were calculated by randomly choosing one of the primes, ‘p’, and multiplying it with the prime that came after it, ’q’, since Pollard Rho has the hardest time factoring large primes and primes that are close. I was not able to run multiple tests as I had hoped since the serial factorizations took very long. The parallelized version of Pollard Rho blew through all 50 semiprimes within seconds while the serial factorizations took literally hours and was not able to finish. I’ve attached the output in “output.txt”, the primes list is “prime32.txt”, the code is “kernel.cu”, and I’ve attached some screenshots here as well. I’ve attached the full Visual Studio folder if you wanted to compile it on your system.



