Outline for paper

1. Desire for parallelizing multiplication
    - a. Further optimize fast multiplication algorithms
        - i. Downsides of long multiplication
            1. Big O is slow O(n^2)
        - ii. Karatsuba’s algorithm
            1. First major improvement to multiplication
            2. Splitting up factor into smaller parts
            3. Significant big O decrease
        - iii. Brief mention of Toom-Cook and Fourier transform (FFS algorithm)
    - b. Large Numbers are typically used for
        - i. Cryptography (generating really large numbers)
        - ii. Cosmology (measuring really large astronomical values)
        - iii. Important for solving some math problems where large numbers need to be checked, such as primes and 3n+1 conjecture
2. Difficulty of parallelizing multiplication
    - a. Splitting up factors
        - i. Must divide string first (use of hex)
    - b. Bit shifting
        - i. Takes up some of the cost that would be otherwise be saved from calculating individual partial sums
3. How we did our naive pthread program
    - a. Use of the GNU library
        - i. Mpz_t struct
            1. Can store an extremely large integers
    - b. Give each task a part of one factor
        - i. Calculate the substring
    - c. Global of other factor
        - i. mpz_t _init_set_str
    - d. Compute partial sum
        - i. Mpz_mult
    - e. Bit shift over correct number of bits
        - i. Mpz_mult_2exp function
    - f. Add all values together(in mutex)
        - i. Mpz_add
    - g. Timing results (how it compares at various values)

Bibliography

Granlund, Torbj¨orn. “GNU MP 6.2.1.” The GNU Multiple Precision Arithmetic Library. Free Software Foundation. Accessed November 23, 2021. https://gmplib.org/manual.