PARALLEL IMPLEMENTATION OF PRIME FACTORIZATION

BY

- MELISSA IVIE

(MELISSA.IVIE@USU.EDU)

- SHAWN WAGNER

(SHAWN.WAGNER@USU.EDU)

- SRIYA GORREPATI

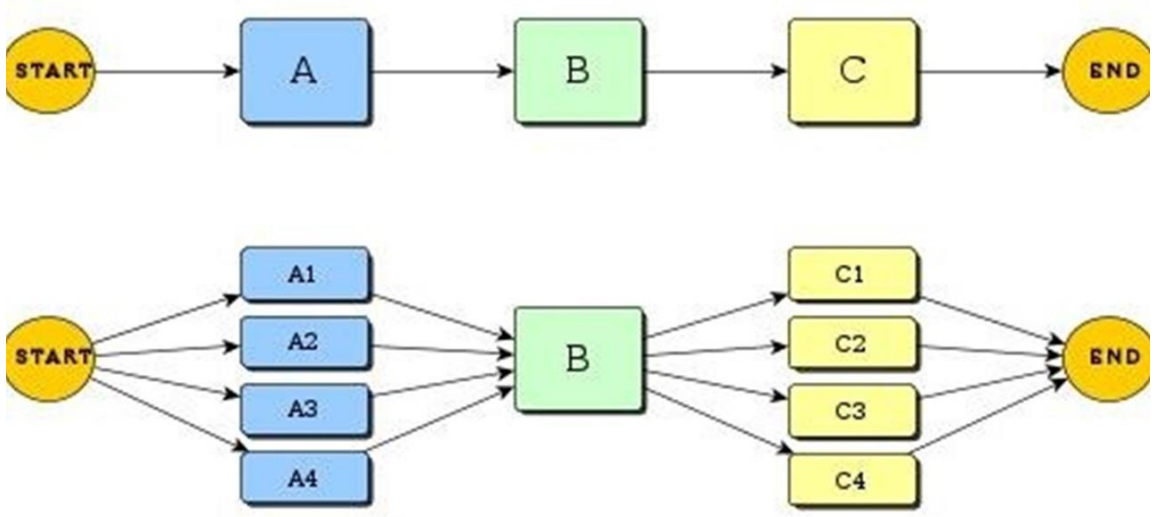
(A02370648@USU.EDU)

INTRODUCTION

- Prime numbers are key components in data encryption/decryption.
- Working with large primes has become important in many security methods.
- Usage of parallel programming improves the performance and efficiency of base prime computations.
- In our project we use the non-parallel method to compare the parallel implementations.

THESIS STATEMENT

- Our thesis is that a parallel implementation of prime factorization will demonstrate an improved performance and efficiency when compared to non-parallel execution methods.



Approach

- Sequential Algorithms
  - Basic
  - Euler
  - Rho

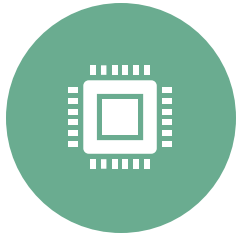
Naïve Parallel Implementation

UTILIZES THE BASE ALGORITHM FOR PRIME FACTORIZATION, ALSO REFERRED TO AS THE NAÏVE PRIME SOLUTION.

PARALLIZED USING MPI_SEND() AND MPI_RECEIVE() COMMANDS

TOTAL NUMBER OF PROCESSORS IS DETERMINED AND USED TO COMPUTE DELTA THAT DIVIDES THE TOTAL RANGE OF VALUES INTO SEGMENTS

CALCULATED RANGE SEGMENT SENT TO EACH PROCESSOR COMPUTES A LIST OF ALL PRIMES WITHIN THE GIVEN SET



ONCE THE END OF THE RANGE IS REACHED, THE CALCULATED PRIMES ARE RETURNED TO THE MASTER PROCESSOR AT RANK 0.

Sieve of Eratosthenes

- Lists numbers from 2 to the desired range, n
- Examine values in the number list starting from the first value, p, and proceeding sequentially through the numbers until p >= √n, all multiples of p are eliminated
- Once p has executed all iterations the number list will be comprised solely of the prime numbers within the range.

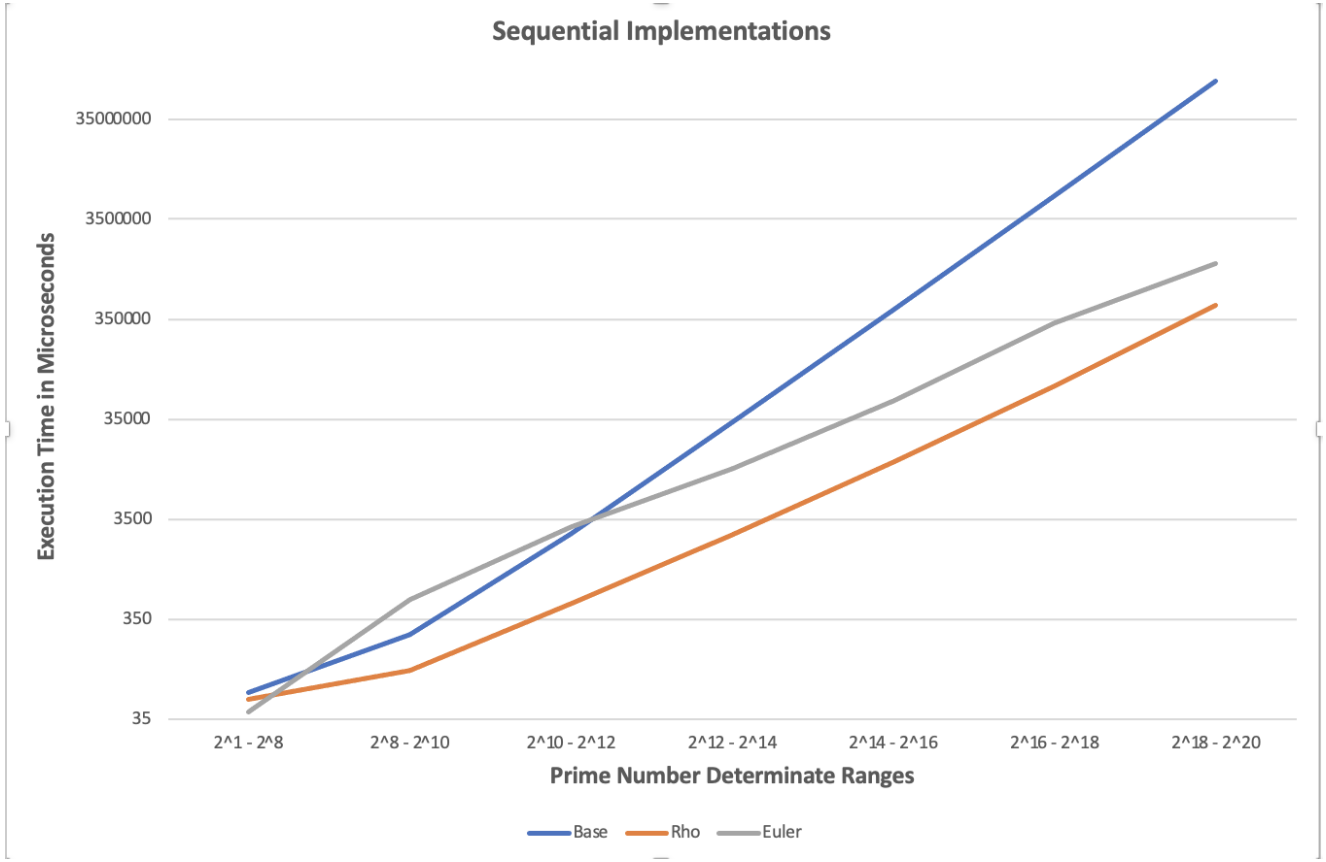
Experiment/Simulation

- Ranges tested in powers of 2: 2^8, 2^10, 2^12, 2^14, 2^16, 2^18, 2^20
- Number of processors used in powers of 2: 2, 4, 6, 8, 16, 32, 64
- Simulation results are a measurement of the elapsed execution time in microseconds (1000000 * seconds)
- Baseline result is measured using a singular processor execution
- Each combination of range and number of processors was tested 10 times recording the average duration as the results

RESULTS OF EXPERIMENT

OVERALL FINDINGS INDICATE THAT RHO IS THE MOST EFFICIENT OF THE SERIAL METHODOLOGIES.

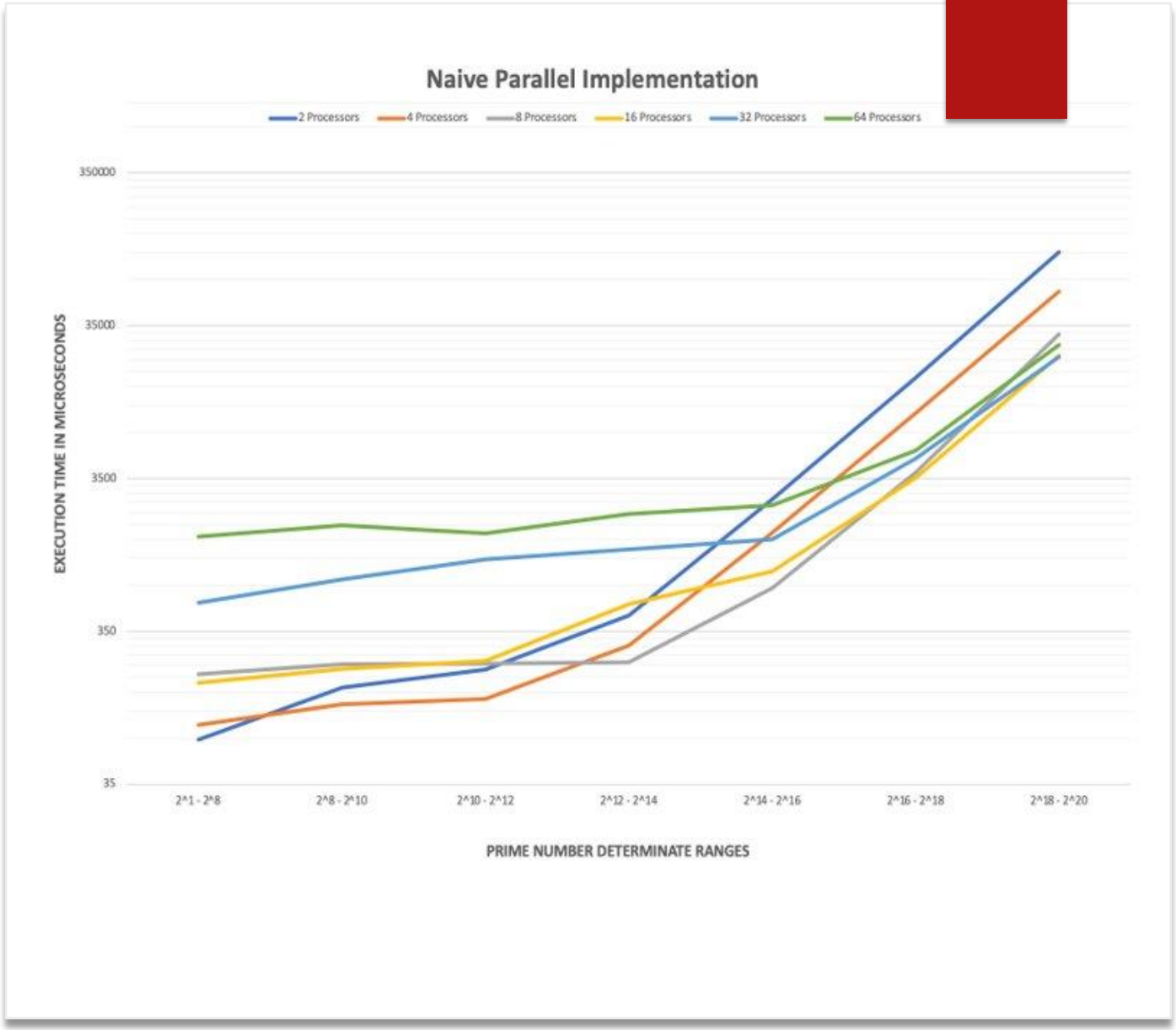
PARALLEL IMPLEMENTATIONS ARE MORE EFFECTIVE FOR HIGHER INTEGER VALUES, EXPERIENCING AS MUCH AS A 86% EFFICIENCY INCREASE. HOWEVER, THERE ARE LESS EFFECTIVE FOR LOWER DIGIT RANGES.



RESULT- SEQUENTIAL IMPLEMENTATIONS

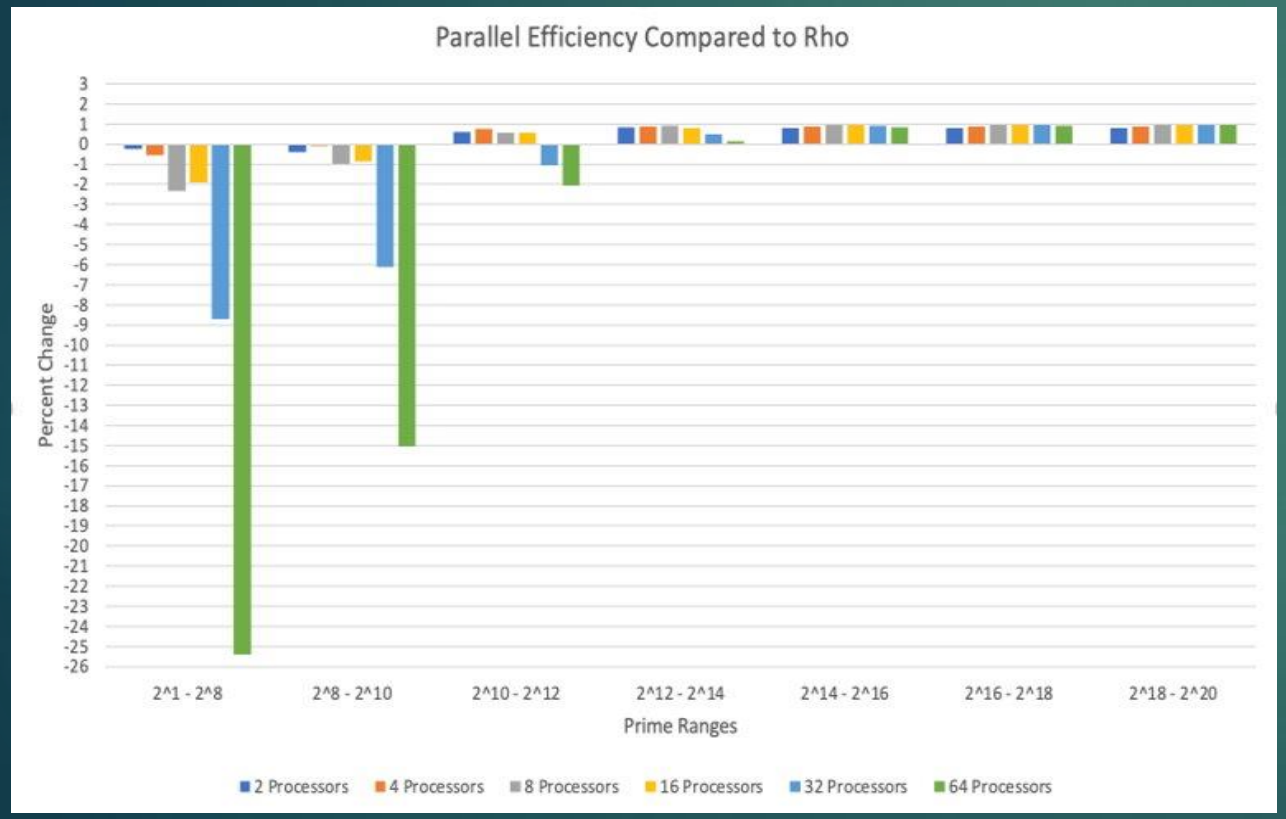
PARALLEL NAÏVE IMPLEMENTATION

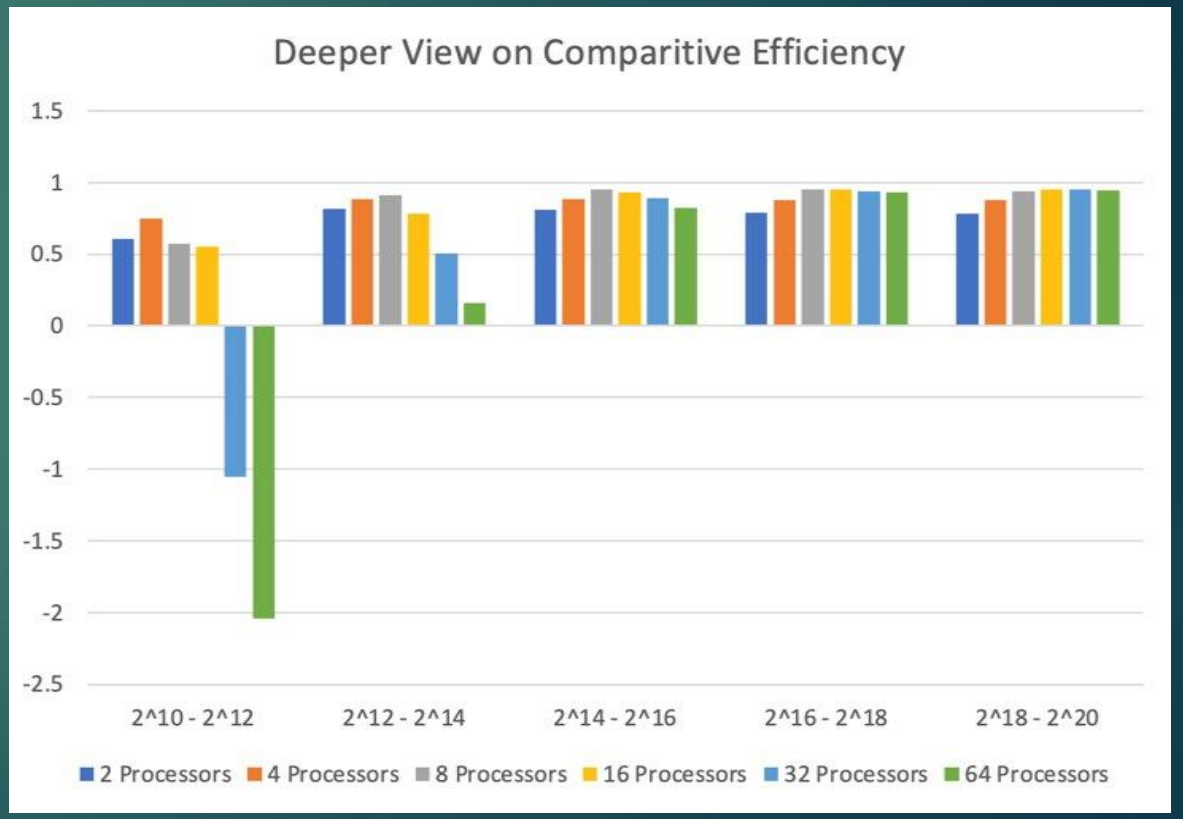
- Key differences compared to Serial approach
  - Less steep slope
  - Higher duration at lower scale
  - Higer degree of Variance

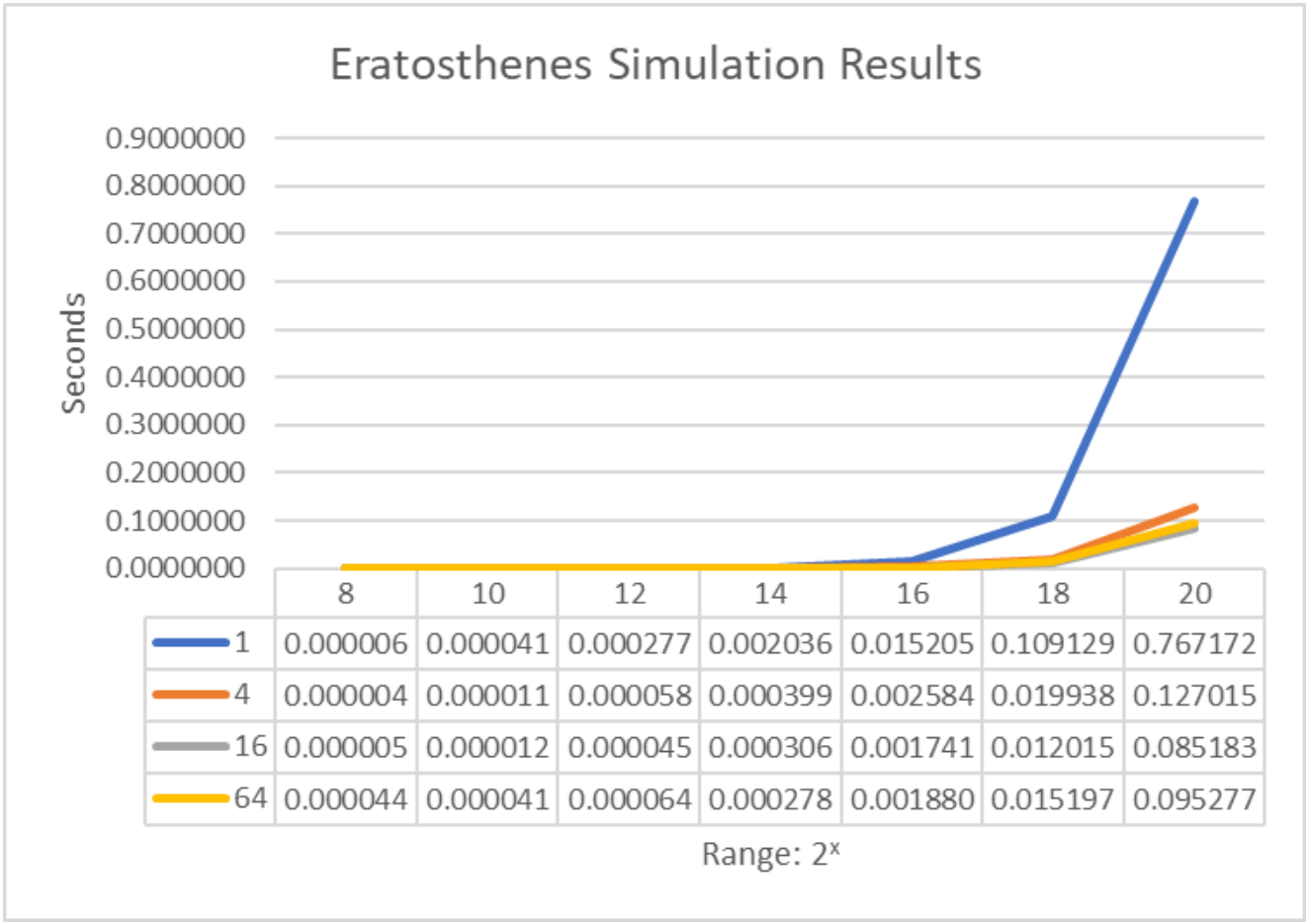


Efficiency Comparison

- Examining the improvement times of parallel applications by measuring the execution times as a percent change of the Rho algorithm results.







	8	10	12	14	16	18	20
1	0.000006	0.000041	0.000277	0.002036	0.015205	0.109129	0.767172
4	0.000004	0.000011	0.000058	0.000399	0.002584	0.019938	0.127015
16	0.000005	0.000012	0.000045	0.000306	0.001741	0.012015	0.085183
64	0.000044	0.000041	0.000064	0.000278	0.001880	0.015197	0.095277

Sieve of Eratosthenes

CONCLUSION

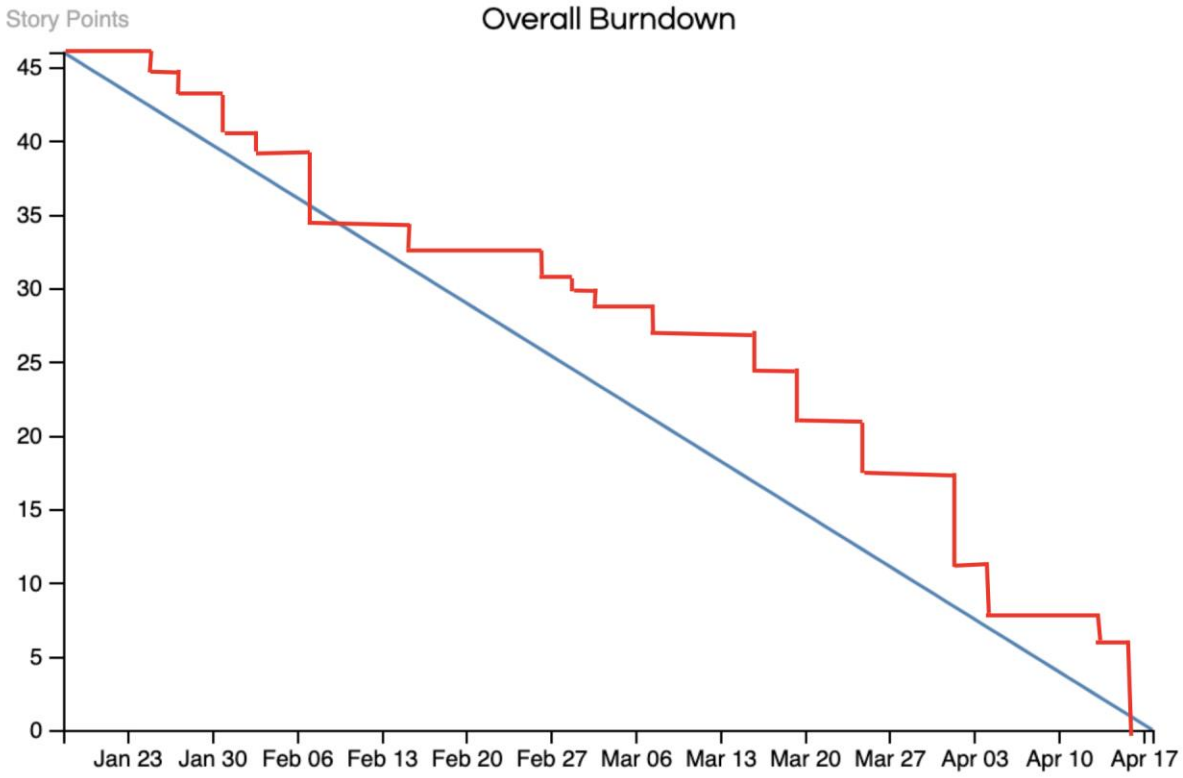
The Parallel implementation does demonstrate improved efficiency in prime factorization for values above 4 digits.

However, at the lower levels from 2 to 2^10, serial methods are more effective.

Increased parallelism with a higher quantity of threads does not directly translate into lower durations until the 2^16 value range.

LIST OF TASKS

Creation of Jira Board Story Points: 1	Overleaf Project Template Story Points: 1	GitHub Repository Creation Story Points: 2	Topic Research Story Points: 1	Introduction Section Story Points: 5	Management Section Story Points: 3	Burn Down Chart Story Points: 3	Detailed Description Story Points: 5
Latex Document Check Story Points: 1	Initial Background Research Story Points: 8	In Depth Background Research Story Points: 8	Linear Prime Factorization Rho Algorithm Story Points: 5	Parallel Prime Factorization Sieve Method Story Points: 8	Linear Prime Factorization with Wheel Approach Story Points: 5	Linear Prime Factorization with Euler Factorization Story Points: 4	Burn Down Chart Generation Story Points: 2
Writing Midpoint Abstract section Story Points: 3	Writing Midpoint Introduction section Story Points: 2	Writing Midpoint Background section Story Points: 4	Writing Midpoint Approach section Story Points: 6	Writing Midpoint Results section Story Points: 1	Writing Midpoint Sprint Tasks section Story Points: 1	References section Story Points: 1	Testing Serial Methods Story Points: 5
	Testing Naïve Parallel Solution Story Points: 6	Testing Sieve or Eratosthenes Story Points: 6	Writing Final Results Section Story Points: 5	Writing Final Conclusion Section Story Points: 3	Writing Final Future Work Section Story Points: 2	Create Final Presentation Story Points: 4	



Burndown Chart