

**National University of Computer and Emerging Sciences FAST**

**Project Report**

Course: **Parallel and Distributed Computing**

Instructor: **Syed Faisal Ali**

Semester: Fall 2023

Project Title: **Numerical Computing Methods Analysis with OpenMP**

Group Members

|  |  |  |
| --- | --- | --- |
| Sno | ID | Name |
| 1 | 21K3323 | Muhammad Shaheer |
| 2 | 21K3161 | Muhammad Ahmed |
| 3 | 21K4556 | Muhammad Anas |

**Introduction**

**Problem**

While numerical computing methods are powerful tools, the computational demands of large-scale problems can be time-consuming. The need for efficient solutions becomes evident, prompting the exploration of parallel implementations. The goal is to execute numerical computing methods concurrently on multiple processing units or cores, leading to improved efficiency and reduced computation time.

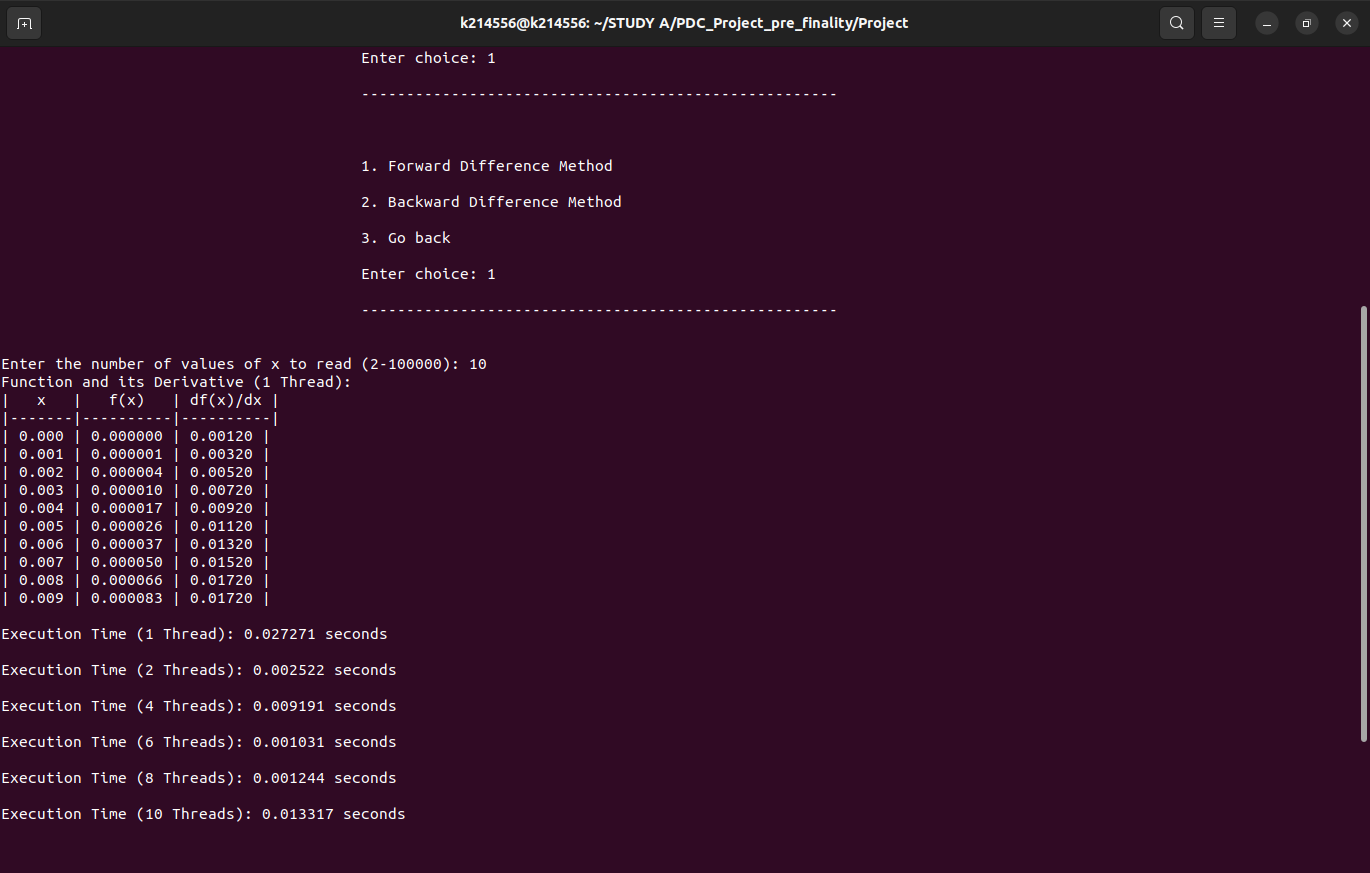
**Algorithm**

The project focuses on implementing the following numerical computing methods

**Numerical Differentiation**

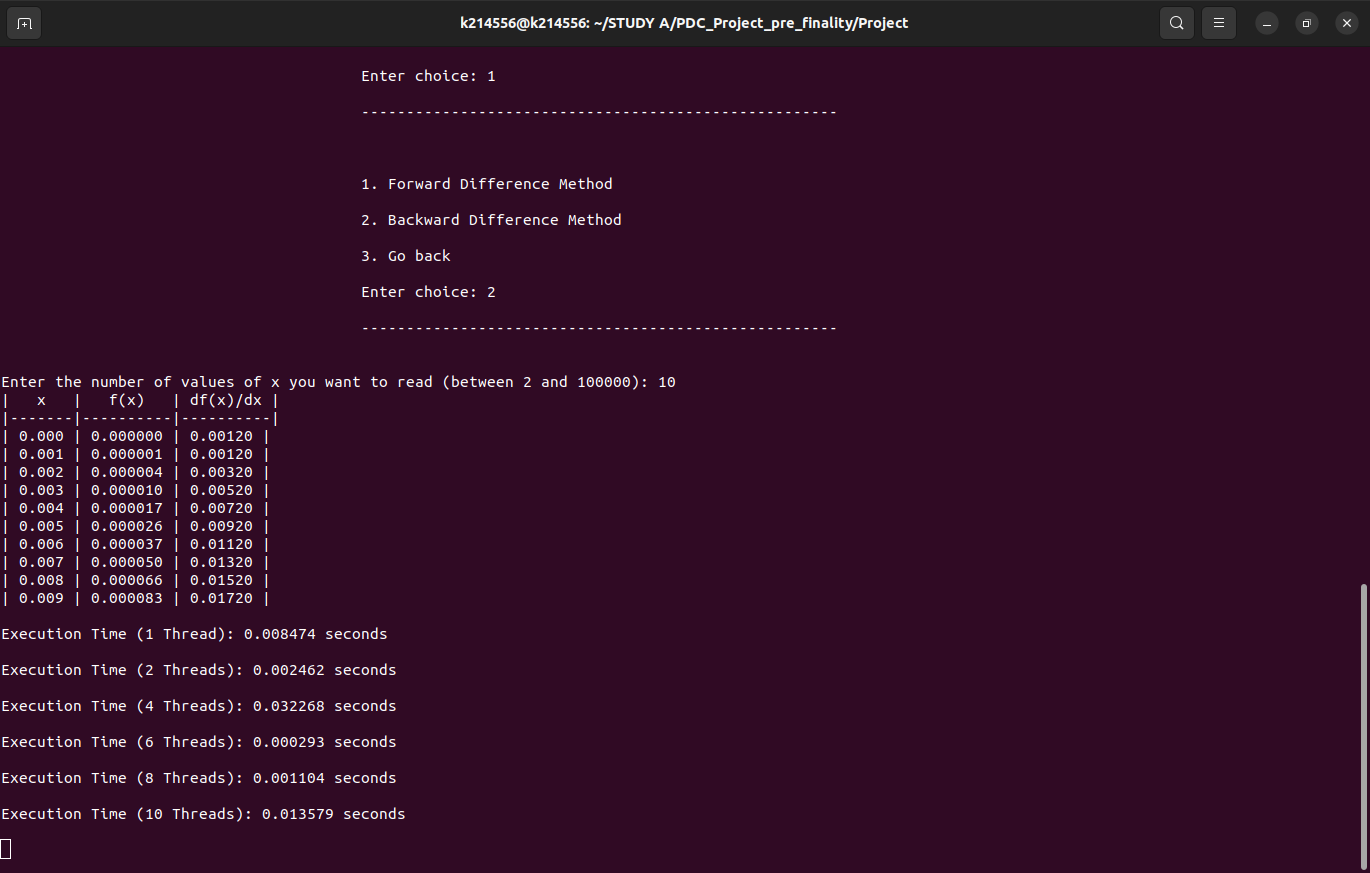
**Forward Difference Method**

The forward difference method is a numerical technique used to approximate the derivative of a function at a particular point. It computes the derivative by considering the function values at the given point and a small increment h. In a parallel setting using OpenMP, this method involves dividing the range of function evaluations among threads to simultaneously calculate the differences and derivatives across different intervals.



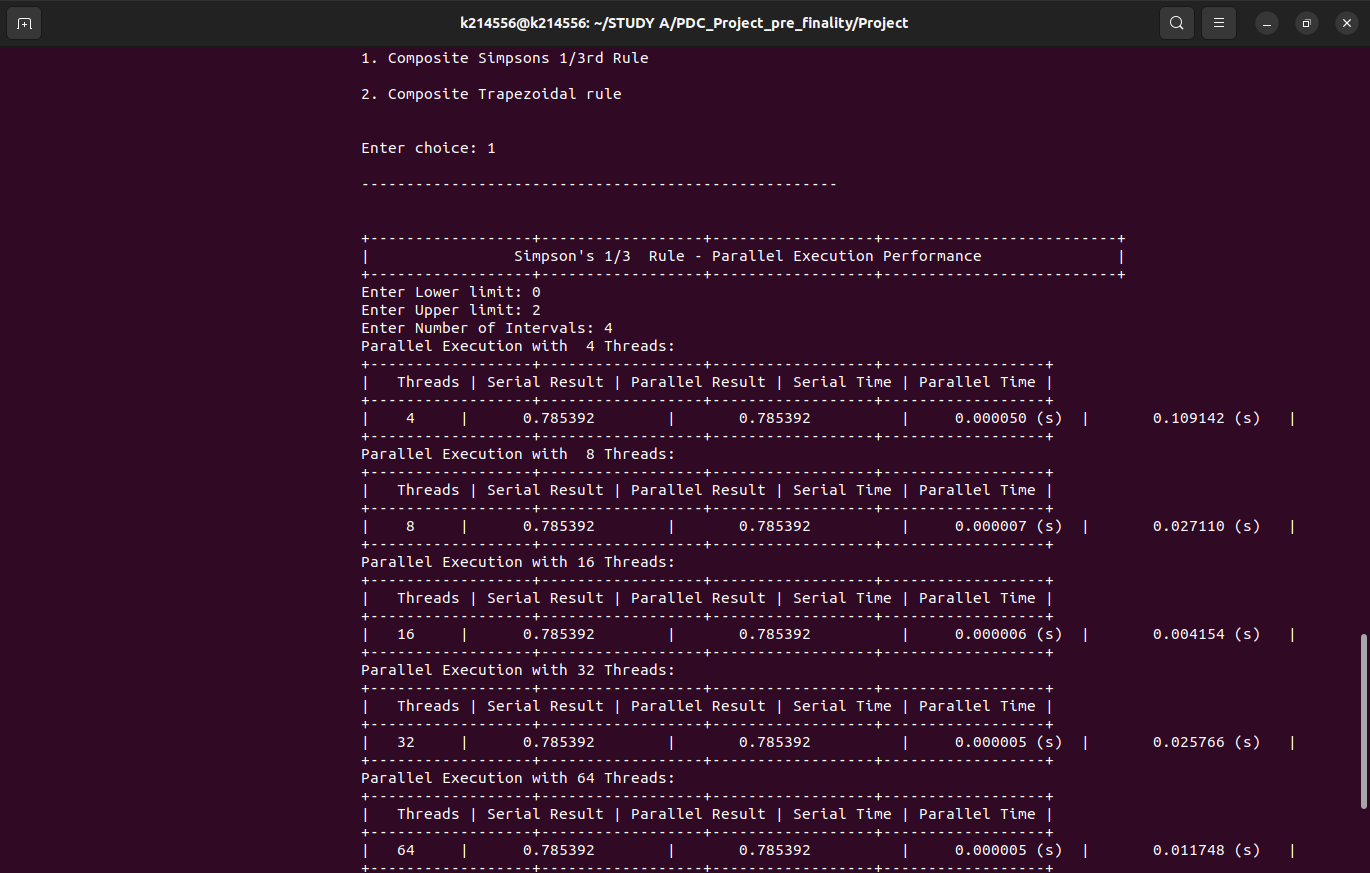
**Backward Difference Method**

Similar to the forward difference method, the backward difference method approximates the derivative of a function but uses backward-pointing intervals. In parallelization with OpenMP, threads can be utilized to compute backward differences concurrently across distinct intervals, enhancing computational efficiency.



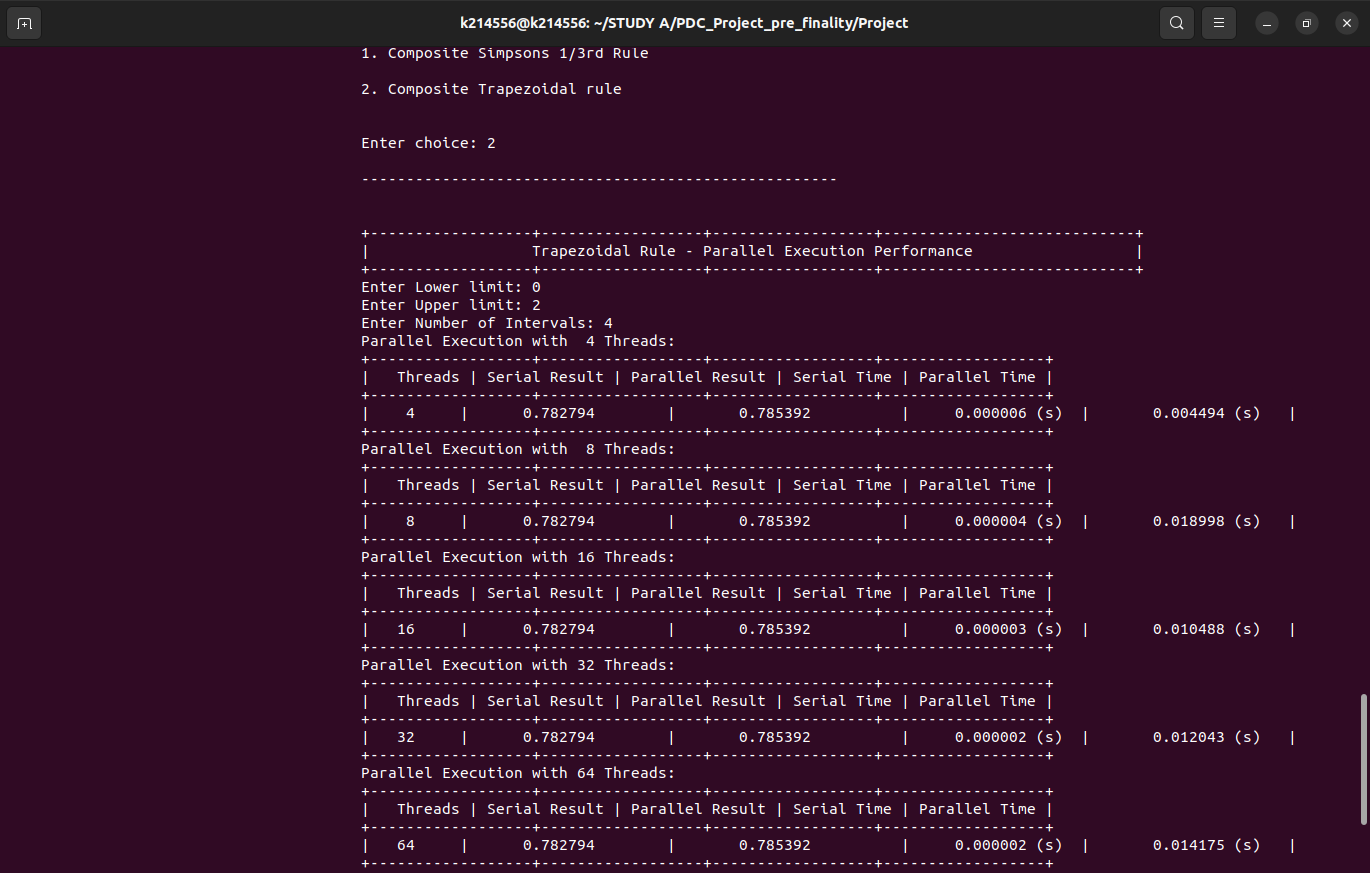
**Numerical Integration**

**Simpson's Method**

Simpson's method is a numerical integration technique that approximates the definite integral of a function by using quadratic polynomials. It divides the interval into smaller segments and employs weighted averages of function values at these points. In OpenMP, this method can benefit from parallelism by assigning segments to different threads for simultaneous computation, accelerating the integration process.

**Trapezoidal Method**

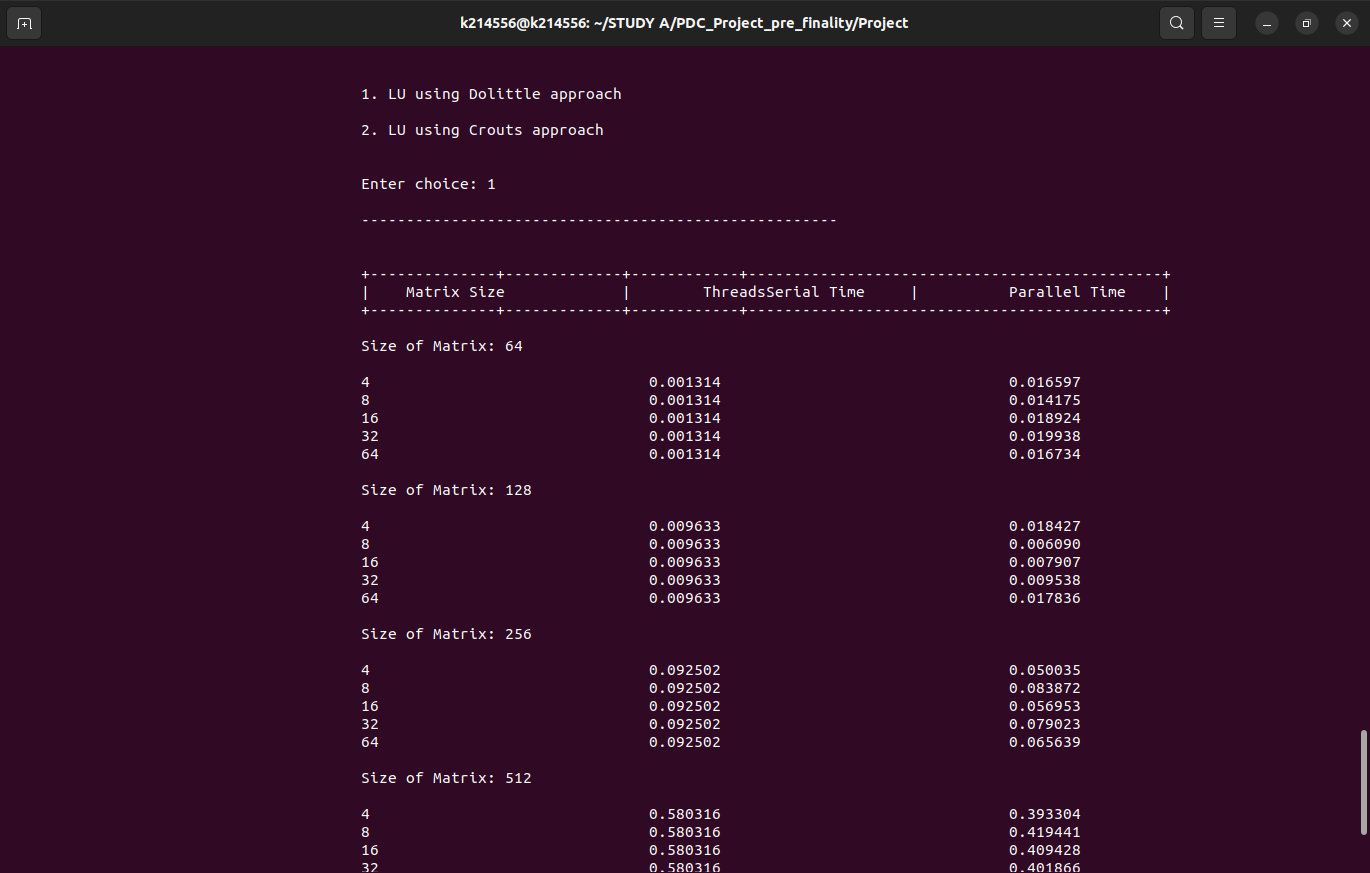
The trapezoidal method estimates the definite integral by approximating the area under the curve using trapezoids. It divides the interval into smaller subintervals and computes the area for each trapezoid. With OpenMP, this method can be parallelized by assigning segments to multiple threads, allowing for concurrent calculation of trapezoidal areas, thereby speeding up the integration process.

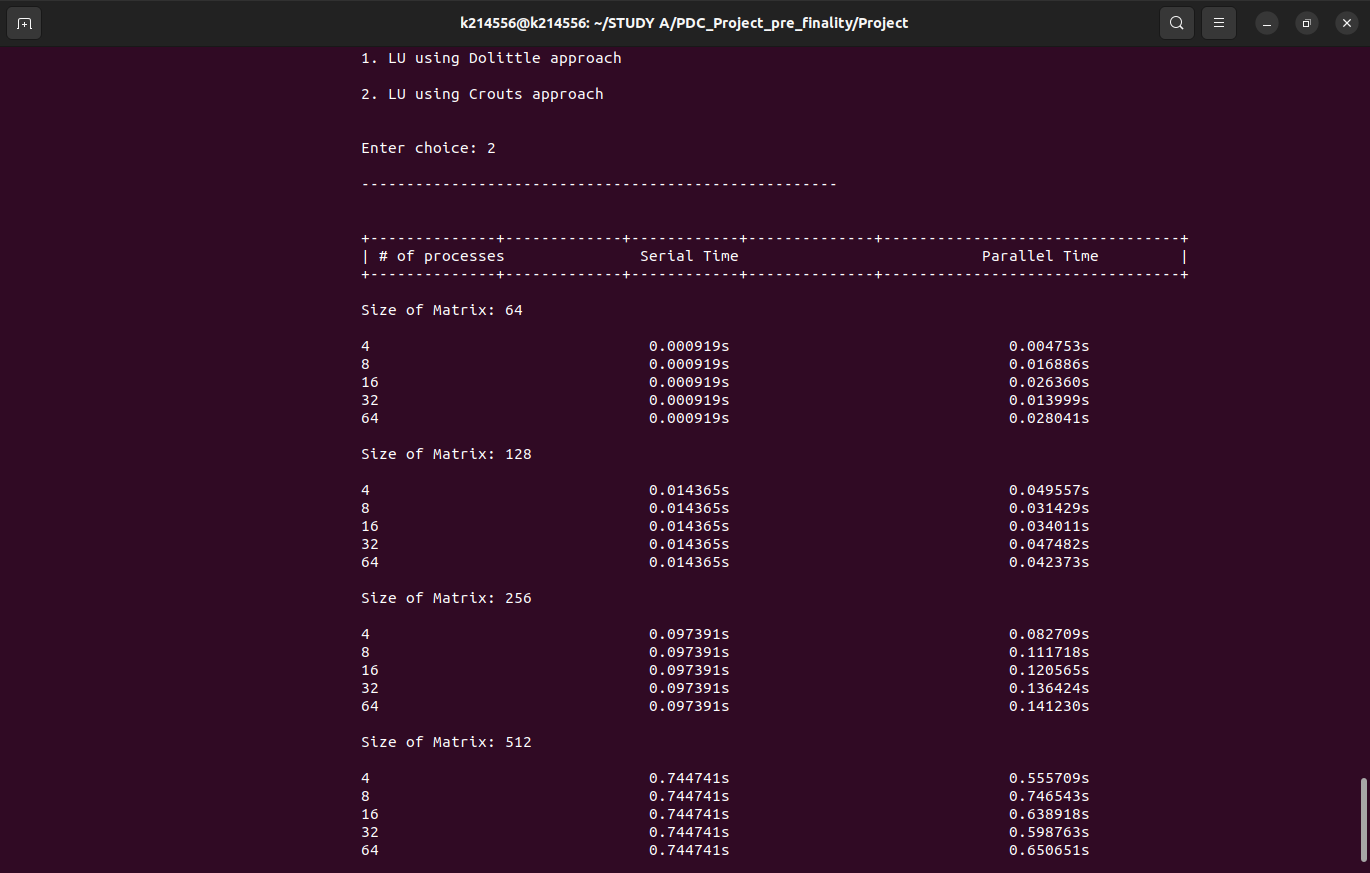
****

**Solving Linear Systems**

**LU Decomposition (Doolittle and Crout)**

LU decomposition is a method used to solve linear systems of equations by decomposing the coefficient matrix into lower and upper triangular matrices. Doolittle and Crout are two approaches to achieve LU decomposition. In a parallel setting with OpenMP, decomposition involves breaking down the matrix operations into parallel tasks, distributing the computation of matrix factors among threads for efficient computation.

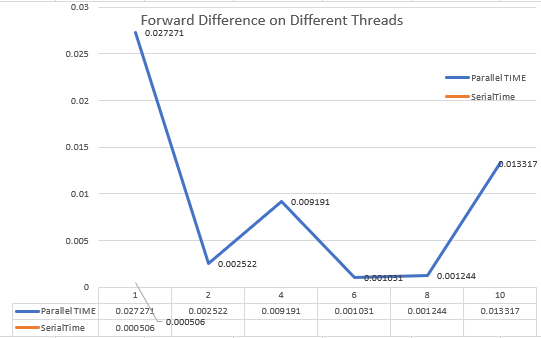




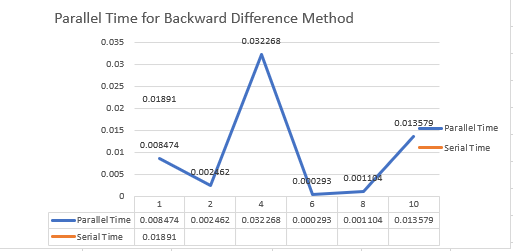
Each of these methods demonstrates varying computational complexities and advantages when parallelized using OpenMP in C on the Ubuntu operating system. Parallelization aims to optimize execution time by leveraging multiple cores or threads for simultaneous computations, thereby enhancing the overall performance of these numerical algorithm

**Serial and Parallel Execution Visualization through Graphs:**

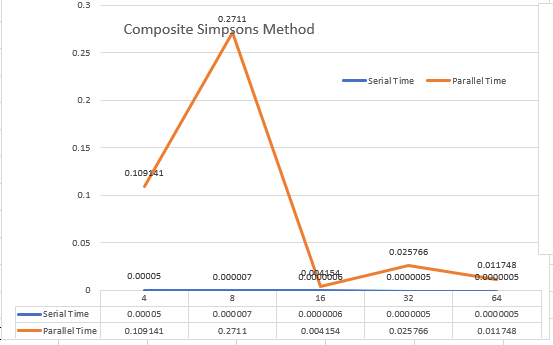
**Forward Difference:**

****

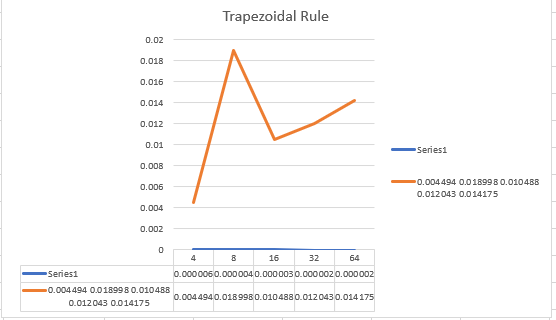
**Backward Difference:**

****

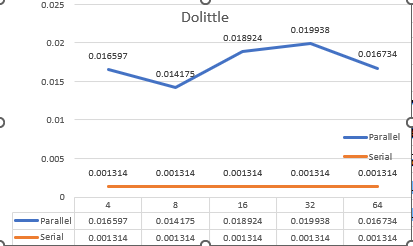
**Simpsons Method:**

****

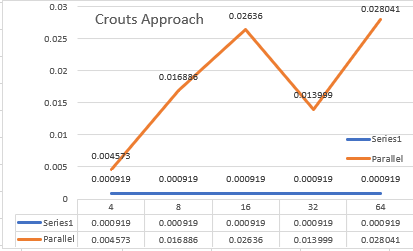
**Trapezoidal Method:**

****

**Dolittle Method:**

****

**Crout Method:**

****

**Specifications of the System Performed:**

|  |  |
| --- | --- |
| Core | I7 |
| Generation | 11TH Generation |
| Number of Cores | 4 |
| Threads | 8 |
| Speed | 3.8ghz |

**Conclusion:**

In summary, the project's exploration of serial and parallel methods using OpenMP has not only demonstrated the advantages of parallelization in numerical computing but has also provided valuable insights into optimizing computational tasks for improved efficiency and scalability. The findings underscore the practical relevance of leveraging parallel processing techniques to address increasingly complex computational challenges in various domains.

Top of Form