Assignment1: Trapezoidal Rule using MPI

Objective

- Implement the trapezoidal rule using MPI to estimate the integral of given functions across multiple processes.
- Measure and analysis execution time using `MPI_Wtime()` and perform scalability and precision tests.

Introduction

1. Base Code: You have been provided with a basic implementation of the trapezoidal rule using MPI for the function $\boldsymbol{f(x) = x^2}$ over the interval $[\mathbf{0}, \mathbf{1}]$**.** The code is similar to what was covered in class. You are expected to modify this code to complete the tasks outlined in this assignment.
2. MPI Timing: The code includes timing functionality using `MPI_Wtime()` to measure the execution time of the MPI program. Please note that the placement of `MPI_Wtime()` is crucial for accurate timing, and you should not change its location within the code.

Instructions

Part 1 – Serial vs Parallel Execution Time Comparison

- **Setup** – For both functions
  - $\boldsymbol{f(x) = x^2}$ over the interval $[\mathbf{0}, \mathbf{2}]$**.**
  - $\boldsymbol{f(x) = x^4 - 3x^2 + x + 4}$ over the interval [0, 2].
- **Task**
  - Run the program with **4096 trapezoids** in two configurations:
    1. **Serial execution**.
    2. **Parallel execution** with **4 processes** using MPI.
  - Measure the execution time for both configurations using `MPI_Wtime()` and compare the results.
- **Deliverables**
  - Report the estimated integral for both functions in serial and parallel execution.
  - Include the **execution time** for each configuration.
  - Discuss the performance differences between the serial and parallel versions.

Part 2 – Scalability Test

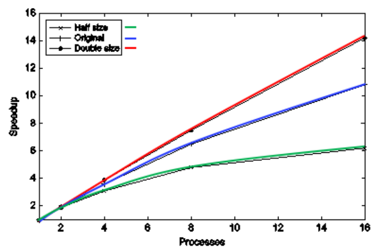
- **Setup** – Run the trapezoidal rule MPI program for both functions
  - $\boldsymbol{f(x) = x^2}$ over the interval $[\mathbf{0}, \mathbf{2}]$**.**
  - $\boldsymbol{f(x) = x^4 - 3x^2 + x + 4}$ over the interval [0, 2].
- **Task**
  - Increase the number of processes and trapezoids to observe how well your program scales.
- **Test Scenarios**
  - Use **1, 2, 4, and 8 processes**.
  - For each number of processes, test the trapezoidal rule with **256, 1024, 4096, and 16384 trapezoids**.
  - Perform these tests for both $\boldsymbol{f(x) = x^2}$ and $\boldsymbol{f(x) = x^4 - 3x^2 + x + 4}$.
- **Metrics**

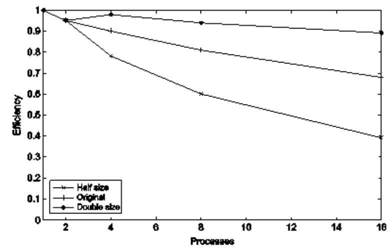
- Measure the **speedup**: $Speedup = \frac{T_{serial}}{T_{parallel}}$
- Measure the **efficiency**: $Efficiency = \frac{Speedup}{Number\ of\ Processes}$

- **Deliverables**
  - Submit the measured **speedup** and **efficiency** for each function.
  - Include **a table** and **graph** showing speedup and efficiency as a function of the number of processes and trapezoids for the functions: $\boldsymbol{f(x) = x^2}$.
  - Include **a table** and **graph** showing speedup and efficiency as a function of the number of processes and trapezoids for the functions: $\boldsymbol{f(x) = x^4 - 3x^2 + x + 4}$.
  - Discuss the performance of the results on both functions.

- **Note** – Refer to **Lecture 5: Performance Metrics** for details on how to compute and analyze speedup and efficiency.
  - **For example,**

	p	1	2	4	8	16
Half	S	1.0	1.9	3.1	4.8	6.2
	E	1.0	0.95	0.78	0.60	0.39
Original	S	1.0	1.9	3.6	6.5	10.8
	E	1.0	0.95	0.90	0.81	0.68
Double	S	1.0	1.9	3.9	7.5	14.2
	E	1.0	0.95	0.98	0.94	0.89





Part 3 – **Precision Test**

- **Setup** – For both functions
  - $\boldsymbol{f(x) = x^2}$ over the interval $[\boldsymbol{0}, \boldsymbol{2}]$**.**
  - $\boldsymbol{f(x) = x^4 - 3x^2 + x + 4}$ over the interval [0, 2].
- **Task**
  - Run the program with **2 processes** and increase the number of trapezoids: **10, 40, 160, and 640**.
  - Compare the estimated integral with the **exact integral values**:
    - For $\boldsymbol{f(x) = x^2}$, the exact value is $\frac{8}{3} \approx 2.6667$.
    - For $\boldsymbol{f(x) = x^4 - 3x^2 + x + 4}$, the exact value is $\frac{42}{5} = 8.4$.
- **Deliverables**
  - Submit a **precision report** comparing your estimated results with the exact values **for both functions**.
  - Discuss how the number of trapezoids affects precision **for both functions**.

Submission Deadline

- 23:59:59 – October 28, 2024.