Parallel Processing Project: Solving the 3D Heat Equation

Aman Hogan-Bailey

University of Texas at Arlington

CSE-5351: Parallel Processing

Contents

Introduction

The purpose of this project was to solve the 3D heat equation using both one MPI process and several MPI processes with different splitting of the 3D grid. The equation in three dimensions is expressed as:

$$\frac{\partial u}{\partial t} = \alpha\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}\right)$$

Part 1: Solving 3D Equation for different Grid Sizes

To solve this equation, I expanded my previous 2D solution into a 3D grid. The solution involves iterating over the grid points and updating the temperature values at each time step.

The computational complexity of solving the 3D heat equation increases rapidly as the grid size increases. The runtime increased about 30x when doubling the runtime as displayed in the table. This means that instead of using [50, 100, 200] for runtime, instead, I had to use [25, 50, 100] for realistic testing. Using a grid size of 200 would've taken 24 minutes to run on the bell cluster, minimum, to run just one MPI process (Table 1).

As expected, increasing the grid size reduces the error by a *factor of 4*, indicating improved accuracy due to the finer resolution. However, this comes at a substantial cost in terms of runtime, which grows exponentially. For larger grids, the computational time becomes exceedingly long.

Part 2: MPI to Parallelize Code

In this part I parallelized our solution using MPI by dividing the 3D grid into smaller subdomains, each assigned to a separate process. This approach allows multiple processes to compute updates simultaneously, significantly reducing the overall runtime. I used ghost cells to handle data exchange between neighboring processes and MPI_Isend, MPI_Irecv, and MPI_Waitall to facilitate non-blocking communication between processes. I printed the binding in the output fiels to ensure hyperthreading isn't being done.

We tested using a grid size of 120 instead of 400 because, as stated previously, the increasing runtime of increasing the grid size, too test the program realistically. A grid size of 400 would've taken 13.2 hours to run on one MPI process.

The results demonstrate a significant reduction in runtime as the number of processes increases. However, we observe a slight increase in error when using more processes. This is likely due to the reduced subdomain sizes and increased communication overhead between processes (Table 2).

Part 3: Scaling Results

The efficiency of parallelization can be quantified by analyzing scaling efficiency. Ideally, increasing the number of processes should proportionally decrease the runtime. However, due to communication overhead and synchronization between processes, real-world performance deviates from this ideal (Figure 1) (Table 3).

The results indicate that while parallelization is effective in reducing runtime, it comes with diminishing returns as the number of processes increases. Beyond a certain point, the overhead of communication outweighs the benefits of additional parallelism.

Conclusions

This project demonstrates the effectiveness of using MPI to parallelize the solution of the 3D heat equation. The results show that parallelization can significantly reduce computation time, especially for larger grids. However, the efficiency of parallelization decreases as the number of processes increases due to communication overhead.

Tables, Figures

Table 1

Errors and runtimes for differing grid sizes. Adjusted since the size of 200 takes 24 minutes.

Grid	Error	Runtime (s)
25	0. 000774	0.033998
50	0. 000193	1.065172
100	0. 000048	45.274510

Table 2

Errors and runtimes for various splits, given a grid size of 120. Used size 120 instead of 400.

Splitting	Error	Runtime (s)
1,1,1	0.005559	126.951661
2,2,2	0.097848	15.463419
2,5,5	0.098015	11.481844

Table 3

Efficiency and Runtimes for various splittings

Splitting	Runtime (s)	Efficiency
1,1,1	126.951661	100.00%
2,2,2	15.463419	12.50%
2,5,5	11.481844	2.00%

Figure 1