MPCS 51087 Project 1, Milestone 2
Serial Advection

Sonia Sharapova

1 Introduction

This report details the process and outcomes for numerically estimating discrete approximate solutions to the advection equation in two dimensions. The core objectives were to introduce parallel computing via OpenMP for performance improvement and to implement dynamic memory allocation for the data matrices. Additionally, the project involved generating visual representations of the simulation data for various time stamps.

2 Methodology

2.1 Parallelization using OpenMP

The existing serial implementation of the numerical simulation, specifically the Lax solver and the initialization routines, was parallelized using OpenMP. Two subroutines for first-order upwind schemes and second-order upwind schemes were added as well.

This involved:

- Identifying independent computation sections within the main loop.
- Integrating pragma omp parallel for directives to distribute iterations among multiple threads.
- Experimenting with different scheduling methods for optimal load balancing.

These parameters are read-in through the command line upon running the c file.

2.2 Dynamic Memory Allocation

To enhance memory management and account for changing array sizes, static allocations of matrices C and Cnext were replaced with dynamic allocations using calloc. This change was critical for handling larger grid sizes and for flexibility in grid dimensions.

2.3 Visualization

The simulation data at various time stamps were visualized to analyze the behavior of the system under different conditions. The C program was modified to write the matrix data into text files, which were then read by a Python script using NumPy and plotted using Matplotlib. Unfortunately, I had last minute file issues so my plots did not turn out as I had hoped, but will

be included in the final report.

2.4 Performance

These tables show that the speed of the program increased with more cores, which is what we expect to see as work is distributed.

2.5 Results and Discussion

The introduction of OpenMP parallelization significantly improved the performance of the numerical simulation. The execution time was reduced, as observed in the comparative analysis between the serial and parallel implementations. The dynamic memory allocation allowed for efficient handling of varying grid sizes, thus enhancing the program's flexibility.

Cores	Problem Size (NXN)	Grind Rate ($\frac{cells}{sec}$)	Parallel Speedup
1	3200×3200	77267769.643	1
2	3200×3200	132357986.362	1.7129
4	3200×3200	204135299.403	2.6419
8	3200×3200	273052508.698	3.5338
16	3200×3200	289722076.723	3.74958
32	3200×3200	331961695.503	4.2962
64	3200×3200	262468833.653	3.3968

Table 1: Performance of Lax Method for specified core counts and problem size. All experiments should use $L = 1.0m$, $u = 1.0\frac{m}{s}$, $v = 1.0\frac{m}{s}$, and $T = 1.0s$ with $\delta x = \frac{L}{N-1}$, $\delta t = 0.5\frac{\delta x}{\sqrt{u^2+v^2}}$ You are free to fill in the empty core count cell with any value greater than 32 that you are able to access.

Figura 1: Plots.

Cores	Problem Size (NXN)	Grind Rate ($\frac{cells}{sec}$)	Parallel Speedup
1	200×200	78990769.248	1
2	200×200	134832561.180	1.707
4	200×200	208297773.768	2.637
8	200×200	288493737.026	3.652
16	200×200	254251765.912	3.219
32	200×200	280422343.268	3.550
64	200×200	3773305.128	0.0477

Table 2: Performance of Lax Method for specified core counts and problem size. All experiments should use $L = 1.0m$, $u = 1.0\frac{m}{s}$, $v = 1.0\frac{m}{s}$, and $T = 1.0s$ with $\delta x = \frac{L}{N-1}$, $\delta t = 0.5\frac{\delta x}{\sqrt{u^2+v^2}}$ You are free to fill in the empty core count cell with any value greater than 32 that you are able to access.

Figura 2: Plots.

Cores	Problem Size (NXN)	Grind Rate ($\frac{cells}{sec}$)	Parallel Speedup
1	800×800	78896063.674	1, rt: 18.569
2	1600×1600	133084556.170	1.619, rt: 86.984
4	3200×3200	204127979.205	2.642, rt: 453.889
8	6400×6400	14182.868124	rt:
16	12800×12800		
32	25600×25600		

Table 3: Performance of Lax Method for specified core counts and problem size. All experiments should use $L = 1.0m$, $u = 1.0\frac{m}{s}$, $v = 1.0\frac{m}{s}$, and $T = 1.0s$ with $\delta x = \frac{L}{N-1}$, $\delta t = 0.5\frac{\delta x}{\sqrt{u^2+v^2}}$

Figura 3: Plots.