Distributed Cellular Simulation with Dynamic Load Balancing

By Gokul Chaluvadi

Abstract— **My project proposal for High-Performance Distribution Systems involves designing and implementing a distributed cellular simulation framework. This framework ideally should be capable of dynamically balancing the computation workloads across multiple nodes. Examples of cellular simulations are things like wildfire spread, epidemic diffusion, fluid flow, and many more. These often suffer from uneven workload distributions over time. My goal is to apply message passing and adaptive load redistribution to achieve efficient scaling and improved resource utilization.**

I. INTRODUCTION

In many areas of science and engineering, we have implementations of applications such as epidemiological modeling, physics-based diffusion, and many more on cellular automata to represent the interactions across a spatial grid. By default, these are naturally parallel since each cell depends on its immediate neighbor. Static domain decomposition is when the grid evenly divides computing across processes. This becomes inefficient when the computation evolves unevenly across the grid. Dynamically load balancing can mitigate it by reallocation the computation regions at run time based on the activity levels.

The main goal for this project is to measure the scalability, the communication overhead, and the impact on performance. The goal is to implement a cellular automation using MPI. Each process would manage a subregion of the global grid and would exchange data with the neighboring ranks. The simulation would involve incorporating a simple wildfire propagation model to test the framework under different workloads.

II. IMPLEMENTATION

The framework/system will be implemented in C/C++ with MPI. The baseline will use a static domain decomposition. Each MPI rank will simulate a fixed grid portion. Dynamic load balancing will be used to periodically measure each rank's computational intensity (the number of active "burning" cells) and migrating partial grid segments to neighboring ranks. To make the simulation more efficient, I will be using a nonblocking MPI communication so that data can be transfered between processes while the computation continues. I will run benchmarks to compare the performance of static versus dynamic grid partitioning for different grid sizes and activity patterns.

III. RELATED WORK

One of the papers combined cellular automata with fire spread equations to predict forest fire propagation accurately. They talked about how local CA rules can capture complex phenomena like wind effects, fuel properties, and terrain interactions. Another paper talked about the challenges of cellular automata by implementing load balancing strategies that efficiently spread work across multiple computers. One focuses on accurate simulation of natural phenomena, while the other focuses on efficient computational distribution. The goal for this project is to combine both ideas. It applies dynamic load balancing to cellular simulations where the computational work is unevenly distributed over time and space. For example, in wildfire simulations. Most computation is done where the fire is actively spreading, while static partitioning allocates equal resources to inactive regions. By dynamically redistributing grid partitions based on where computation is actually needed, this project's goal is to have computational efficiency in long-running cellular simulations without reducing accuracy.

IV. EXPECTED OUTCOMES

The expected outcomes include:

- A functional distributed cellular simulation with both static and dynamic load balancing modes.
- Performance evaluation showing scaling behavior across varying process counts.
- Visualization and analysis of workload distribution over time.

V. CONCLUSION

This project's goal is to design and implement a distributed cellular automata simulation that is capable of dynamically balancing computation loads across multiple nodes. The system would ensure efficient utilization of both CPU and network resources in a cluster environment. The proposed approach will be implemented using MPI, OpenMP or CUDA for intra-node parallelism. This project contributes a reusable framework that can be extended to real-world scientific and biological simulations requiring adaptive parallelism.

REFERENCES

[1] X. Sun, et al., "A Forest Fire Prediction Model Based on Cellular Automata and Machine Learning,"

[2] R. Hofestädt, et al., "Simulation of Load Balancing with Cellular Automata,"

[3] "A Dynamic Approach to Load Balancing in Cloud Infrastructure: Enhancing Energy Efficiency and Resource Utilization,"

[4] M. Hasan and A. Kliks, "Enhancing Wireless Network Efficiency with the Techniques of Dynamic Distributed Load Balancing: A Distance-Based Approach,"