Using HPX to improve the scalability of Simulations

Jonathan Parziale¹, Hartmut Kaiser², Zach Byerly³, Bryce Adelstein-Lelbach⁴

College of Staten Island¹, Louisiana State University Center for Computation and Technology, LSU Department of Physics & Astronomy³

Introduction

For a long time scientists have modeled hydrodynamics simulations on super computers using parallel techniques such as OpenMP or MPI. This works fine until they try to scale these simulations to use bigger machines, with increased resolution. Using these techniques to scale simulations such as hydrodynamics, takes a lot of time and it is not easy to implement.

The Problem

The issue is that these simulations imply global barriers, that is what makes them hard to scale. Furthermore as we try to scale these simulations with the old techniques, we increase effects that are known as S.L.O.W, and we promote barriers.

Time for Execution of 500000 Threads (Artificial Work: 100ps)

![Graph comparing performance impacts](Image)

The S.L.O.W. effects

- Starvation – non-consistent flow of work to be done, in order to maintain high utilization of resources.
- Latency - Time/distance delay to access remote resources.
- Overhead - An increase in the amount of work needed to manage the environment.
- Waiting for contention resolution - Delays due to lack of availability of oversubscribed shared resource.

What is HPX

- It is a parallel run time environment
- That was Developed in C++
- It Uses fine grained parallelism
- Provides an asynchronous parallelism
- Utilizes an active global addressing scheme for managing resources.
- Administers programing tools to accomplish the HPX parallel model.

What we worked on

We are working on Software which simulates one dimensional hydrodynamic fluid motion. This was originally written by Zach Byerly, with the help of the stellar group, and written using HPX.

What we added this summer

- Restructured code so it can be more modular making it easy to integrate as a part of HPX.
- Instead of having separate representation of the time steps and the cells, we unified them into one object to better represent the grid structure of the 1d Hydro simulations.
- While restructuring the code we were able to simplify the data dependencies of the cells, therefore being able to utilize the asynchronous functionality of HPX.
- Improved the simulation so that it can run to a physics time of interest, that will be specified by the user.

Results

- Before: Time for the simulation to complete was between 270-300 seconds.
- After restructuring: Time for the simulation to complete was between 120-150 seconds.
- 44%-50% improvement.
- This is due to the fact that HPX was developed to work with code that has well defined I/O.
- Therefore helping expose the Software to utilize parallelism when using HPX.

(Tests above where done using 50 cells and 100 time step simulations)

Acknowledgments

This material is based upon work supported by the National Science Foundation under award OCI-1005165 with additional support from the Center for Computation & Technology at Louisiana State University.

All work in this poster is made possible thanks to the stellar research group and Zach Byerly.

References

https://stellar.cct.lsu.edu/info/publications/