HPX Application: HAD Adaptive Mesh Refinement

The usual suspects
[1] *Department of Physics and Astronomy, Louisiana State University, Baton Rouge, LA 70803-4001*
[2] *Center for Computation and Technology, Louisiana State University, Baton Rouge, LA 70803-4001*

(Dated: January 30, 2010)

Details efforts at creating an adaptive mesh refinement (AMR) toolkit based on HPX.

PACS numbers:

I. INTRODUCTION

There are many important physical scales which need to be adequately resolved when simulating the orbit and merger of compact astrophysical objects like neutrons stars and black holes. These scales include (1) the individual stars, preferably incorporating some of their internal dynamics, (2) the orbital length scale, (3) the gravitational wave zone, and (4) the location of outer boundaries. The computational demands required to resolve these different physical scales are best met using adaptive mesh refinement.

In the past this challenge has been met using the publicly available computational infrastructure HAD to provide parallel distributed AMR for our codes [1, 2]. As an example, Figure 1 illustrates the resulting mesh structure at a pre-merger stage of a binary neutron star system in our simulations.

Strong scaling results of the HAD toolkit are illustrated for a particular sample problem in Figure 2. We define speed-up as

$$\text{speedup}(n) = \frac{\text{Run time on one processor}}{\text{Run time on } n \text{ processors}}.$$

The results presented are strong scaling results; the global problem size was kept constant while the number of processors varied. Strong scaling tests are problem dependent and vary according to the size of the global problem selected for investigation. For most problems we investigate, strong scaling using the HAD toolkit is sufficient only up to about 256 processors.

The HPX implementation of ParalleX provides a way to eliminate the global barriers which impair the scaling of the HAD toolkit. A 1-D implementation of the HAD AMR toolkit has just been completed; a 3-D implementation is coming shortly. In the next section we outline the key concepts and data structures inside the new HPX based HAD for both 1-D and 3-D implementations.

II. HPX BASED HAD

The HAD toolkit in it's original form contains a global barrier every timestep: no computation can proceed once the global barrier has been reached until every point has reached the same timestep in the simulation. This type of global barrier is not unique to HAD : other major finite difference and finite element based AMR toolkits also contain this global barrier. This global barrier causes major problems when running on large numbers of processors because most processors end up waiting for others to reach the global barrier. Improved load balancing can help reduce this problem but will not solve it. ParalleX enables us to remove all global barriers from the simulation pipeline, including the ubiquitous timestep barrier. The HPX based HAD toolkit has been redesigned in order that all points can proceed computing the next timestep in the simulation without waiting until the neighbor point is ready to do the same.

Three types of mesh objects are introduced to remove all global barriers. The fundamental data structure is a node point. Each node point is autonomous and is communicated. This is in contrast to what is normally done with finite difference AMR codes where the large blocks of node points are treated autonomously and the boundaries of these blocks of nodes points are communicated. See Figure 3.

Acknowledgments: We would like to thank S. Liebling and L. Lehner for stimulating discussions.

[1] Http://www.had.liu.edu/.
[2] S. L. Liebling, Phys. Rev. **D66**, 041703 (2002).

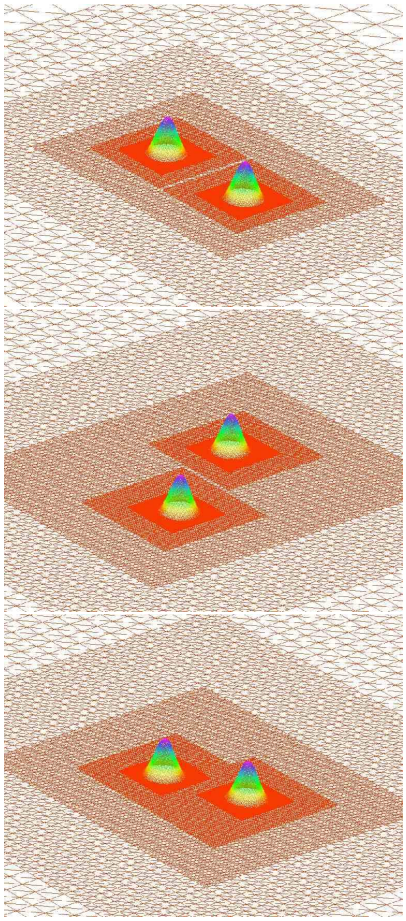
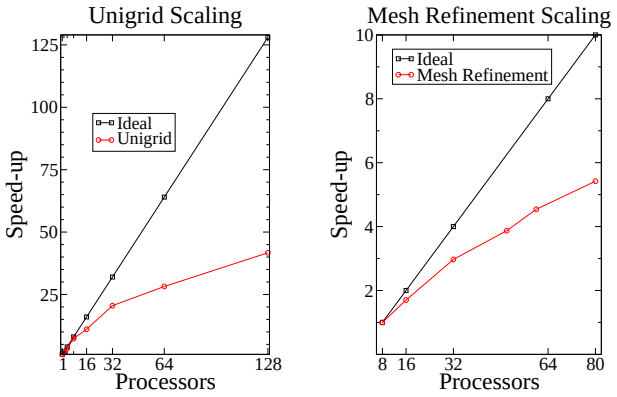
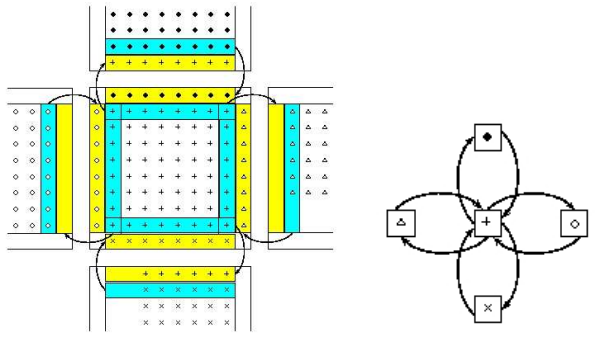


FIG. 1: The AMR mesh structure at times 0, 84, and 500 for the pre-merge stage of the simulation with a highest resolution of 32 points across each star. The simulation had seven different resolution levels depending on the relative error measured locally in the computational domain. Five of those resolution levels are visible here. Simulations were performed on 128 processors.



(a) **(b)**

FIG. 2: This figure shows strong scaling results for single grid and mesh refinement using the MPI based HAD AMR toolkit. A spherical blast wave initial dataset wase run for a fixed problem size as the number of processors is varied. The left frame shows the unigrid strong scaling results, and the right frame includes mesh refinement. For the unigrid scaling, the data were evolved for 80 iterations, and the global grid size was 121^3. For this problem size, the communication overhead begins to overshadow the local process computation on ≥ 64 processors. For the mesh refinement scaling, thirty iterations were performed on a coarse grid of size 81^3 and a single level of refinement. Since the test problem would not fit in memory on a single processor, speed-up was measured using 8 processors as the base value. All tests were performed on an Intel Pentium IV 3.0 GHz cluster with Myrinet.



(a) **(b)**

FIG. 3: Two different approaches to structured mesh based communication: (a) 2-D representation of a typical communication pattern for a finite difference based AMR code. Large blocks of memory are passed to the user for computation where only the boundaries of the blocks are communicated among processors. In this figure, the yellow regions (frequently referred to as ghostzones) are communicated regions originating from blue zones on a distributed memory block as indicated by the arrows. Contrast this with the approach used for HPX based HAD seen in (b). Here each point is communicated giving the simulation the smallest granularity possible. While the amount of communication required in paradigm (b) is substantially more than in (a), those communication costs are reduced by the locality management inherent in HPX. Paradigm (b) is advantageous for eliminating global computation barriers.