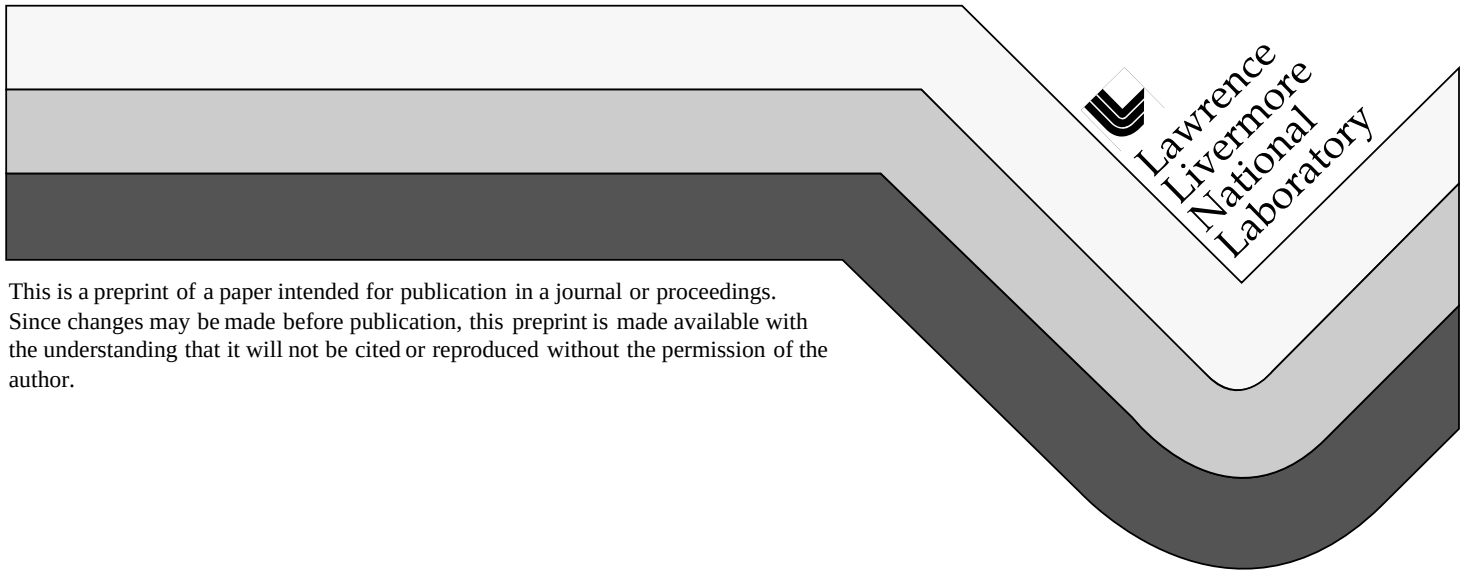
UCRL-JC-132166
PREPRINT

Performance of ALE3D on the ASCI Machines

W. Scott Futral
Evi Dube
J. Robert Neely
Tim G. Pierce

This paper was prepared for submittal to the
Nuclear Explosives Code Development Conference
Las Vegas, Nevada
October 25 - 30, 1998

February 8, 1999



Performance of ALE3D on the ASCI Machines

W. S. Futral III, E. I. Dube, J. R. Neely, T. G. Pierce
Lawrence Livermore National Laboratory

With the anticipated delivery of the ASCI Blue Pacific SST machine approaching, the scaling and performance on large numbers of processors for B Division applications codes have become a matter of considerable interest. Besides the practical impact on users (achievable problem size and run time), the application codes' performance are the ultimate measure of the success of the ASCI machines. The ALE3D code was used to evaluate the performance of the current Blue Pacific Technical Refresh hardware and software in various modes of running. We will present results and analysis of the performance behavior from this study, along with results from other ASCI machines. While gathering statics from user problem runs is easy to do, it is difficult to analyze the variation in performance from problem to problem, or to adjust the problem size consistently for scaling studies. Trivial problems can be used, but may not well reflect the actual performance users can expect. For this study, a series of problems were used that reflect the characteristics of real user problems, but allow for uniform, constant work per processor scaling of problem size and well understood communication characteristics between processors. Additional results for fixed-size problems are presented. Runs were done using different message passing modes (User Space, I.P. protocol), processors per node, and environment settings to investigate the IBM SP2 performance characteristics. (U)

Keywords: hydrodynamics, ASCI, performance

A Brief Introduction to ALE3D

A General Description. ALE3D is a finite element code that treats fluid and elastic-plastic response on an unstructured grid. The grid may consist of arbitrarily connected hexahedra, and the mesh can be constructed from disjoint blocks of elements which interact at the boundaries via slide surfaces. Nodes can be designated as relax nodes and ALE3D will adjust their position relative to the material in order to relieve distortion or to improve accuracy or efficiency. This relaxation process can allow nodes to cross material boundaries and create multi-material elements.

The basic computational step consists of a Lagrangian step followed by an advection, or remap step. In the Lagrangian phase, nodal forces are accumulated and an updated nodal acceleration is computed. The stress gradients and strain rates are evaluated by a lowest order finite element method. A diagonal mass matrix is used. Second order accuracy is obtained with a grid that is staggered in both space and time.

The advection, or remap step allows for either a pure-Eulerian calculation, in which the nodes are placed back in their original positions, or a more complex scheme involving mesh relaxation techniques. This nodal motion or relaxation generates inter-element fluxes which must be used to update velocities, masses, energies, stresses and other constitutive properties. Second-order-accurate schemes are required to perform this operation with sufficient accuracy.

The interaction at slide surfaces can consist of pure sliding in which there are no tangential forces on interface nodes, or the nodes may be tied to inhibit sliding entirely, or a coulomb friction algorithm can be used,. Voids may open or close between the surfaces depending on the dynamics of the problem, and there is an option to allow a block to fold back on itself (single-sided sliding). Where no void is present, the normal forces on either side of the slide surface are accumulated and used to produce a net acceleration of the nodes on the surface consistent with the center-of-mass motion. The ability to remove slide surfaces allows for the flexibility for advecting across these boundaries.

ALE3D is one of the next generation ASCI codes, simulating safety and manufacturing problems. In order for the code to solve these types of problems, new code physics, as seen in Figure 1., must be added to accurately predict the responses to hazard scenarios and manufacturing needs.

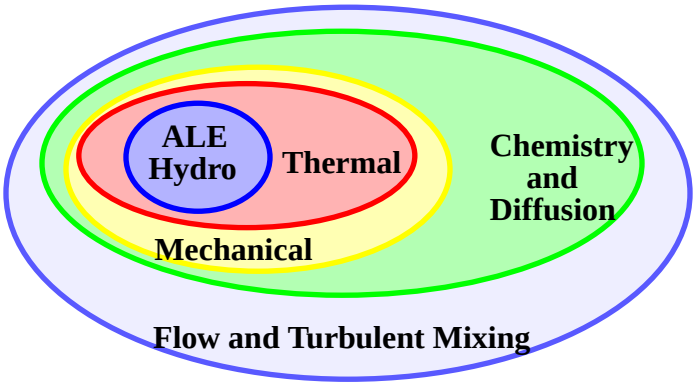


Figure 1. The Physics in ALE3D.

The Parallelization of ALE3D. ALE3D is parallelized across the problem space using domain decomposition. The implementation uses MPI to communicate across processor boundaries, and threads (implemented with compiler directives) with explicit memory copies to communicate on SMP type nodes.

The amount of communication required is dependent on the type of calculation being run. Any problem running without a fixed timestep has one global reduction per cycle to determine the minimum timestep across processors. All other communications referred to here are point-to-point, with data being transferred between domains which are geometrically connected in the problem space.

Problems running explicit hydrodynamics without advection require a single communication in the Lagrange step to collect the sum of the forces at nodes along domain boundaries. Some optional algorithms (e.g. Monotonic Q) require more communication.

Problems running advection require a much greater amount of communication, both in terms of the number of communication points, and the amount of data typically sent. A rough breakdown of the steps performed in the advection are: nodal relaxation; calculation of volume fluxes; identification of mixed elements and interface reconstruction; advection of element centered variables; and momentum (node centered) advection. The amount of communication required for a particular problem is often dependent on both the algorithms chosen by the user and the amount of mixed element data which needs to be communicated near domain boundaries.

Nodal relaxation requires around 7 communication steps per iteration. One communication is required to refresh first order nodal coordinates, while the rest are used to propagate relaxation weights, and to smooth the relaxation near boundaries where relaxation weights differ.

Nodal relaxation at material boundaries will generate new mixed material elements. Describing the layout of these new mixed elements at domain boundaries requires a communication. As with almost all of the communication dealing with mixed elements in ALE3D, this communication is non-symmetric in that a domain sending data to a neighboring domain may not necessarily need to receive data. In addition, the amount of mixed data being communicated cannot be known a priori - requiring a small communication just to provide neighboring domains with this information.

Interface reconstruction is the process of figuring how much of each material in a mixed zone is transferred across each face once the volume fluxes are known. Because ALE3D uses a single-step advection scheme (versus a 3D “sweep” algorithm), the code must be careful not to “overdeplete” a zone by transferring more material out of a zone than exists in that zone. This entire process requires on the order of six communications per cycle.

Advection of element centered variables is a second order calculation, which requires that data in ghost elements up to two zones away from domain boundary elements in the local domain have refreshed data. Transfer of this data is by far the largest communication in the code - as it requires transfer of approximately 20 element centered variables into first and second order ghost elements.

Momentum advection requires several communication steps akin to the communication in the lagrange step which summed the partial force values. In the momentum advection, mass and momentum flux vectors are summed at the nodal boundaries.

Of course, if there are slide surfaces, additional communication will be required, which is described below.

All in all, the amount of communication required to run advection is about 20 times that required in the lagrange step. The amount of time spent in computation in the advection step is approximately five to six times that of the lagrange step. Based on this, we would expect advection problems not to scale as well as “pure lagrange” problems.

In slide surface calculations, the two sides of the surface exert forces on each other, with the result that the net accelerations of the two sides are equal in the direction normal to the surface. In the tangential direction the sides are allowed to slip relative to each other. In order to calculate the net acceleration at a node, information must be available not just for the node, but for a patch of nodes adjacent to it on the opposing side.

This adds complexity to the parallel algorithm, because the adjacency of the two sides constantly changes as the surface slips tangentially. In general, adjacent nodes on the two sides will reside on different processors in the element decomposition, and as the surface slips the processors themselves holding adjacent nodes will change, as shown in Figure 2.

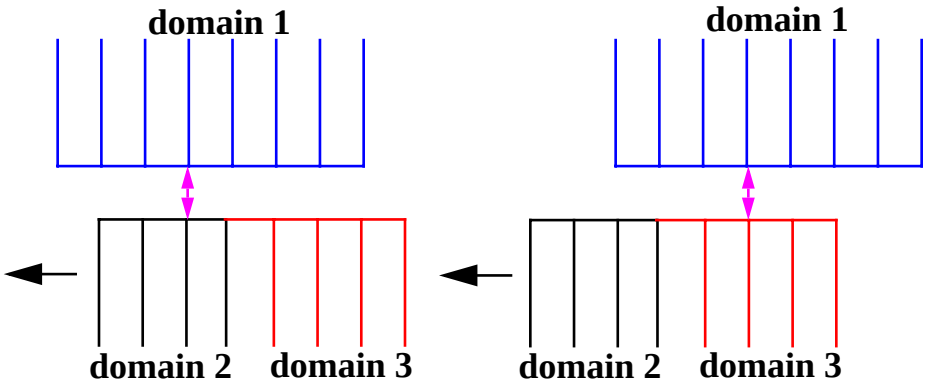


Figure 2. The Dynamic nature of the slide surface decomposition.

. The approach taken in ALE3D is to use a separate decomposition for slide surfaces. Nodes on one side of a surface (the “master” side) are assigned statically to the various processors in a load-balanced manner, and then nodes on the other (“slave”) side of the surface that are currently “close” to the masters on a processor are assigned to the same processor. At the start of a slide calculation, the required data is transferred from the element decomposition to the slide decomposition, with the results transferred back at the end of the slide calculation. As the problem evolves, both the sizes of messages and which processors communicate changes.

In the current implementation, the communication costs of determining which processors communicate and updating the node adjacency information both increase with processor count. Work is currently in progress on a truly scalable algorithm, which will capitalize on the limited amount of tangential movement allowed per timestep.

Scaling Problems

The Cylinder problem. The Cylinder problem is a realistic, yet uniformly scalable test problem. This problem, shown in Figure 3., has four materials, arranged in 6 shells, and all materials are advecting. There are 10080 elements per quarter section of the cylinder, and one section (domain) is a quarter of the cylinder; 4 sections (domains) comprise a complete cylinder segment.

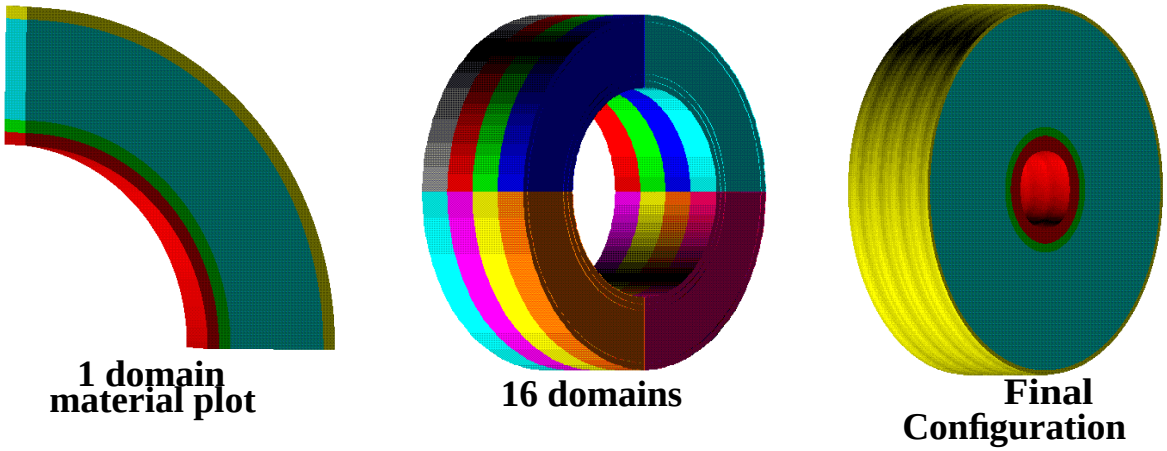


Figure 3. The Cylinder Problem

When the problem is large, interior domains communicate with 8 neighbors. Work and communication are identical for each interior domain.

For measuring performance for this problem, the work load per processor was kept constant. The constant work load problem amortizes communication costs and avoids spurious cache size effects, producing a more accurate measure of parallel efficiency. More cylinder sections (domains) are added to scale up the problem, but keep the work load constant per processor.

The real user wants to know not only how fast the code will execute one physics cycle, but also how performance scales as more zones are added to the problem.

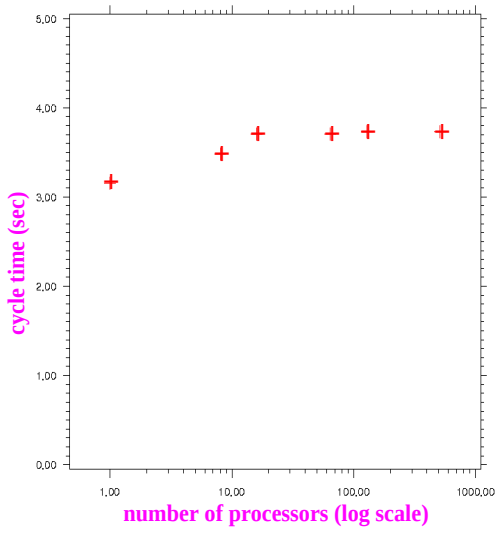


Figure 4. Cylinder problem performance on the Red machine.

The first performance measurement, seen in Figure 4., is for the ASCI Red machine, located at Sandia, Albuquerque. The Red machine shows nearly linear scaling, although it has the slowest processor speed of the ASCI machines.

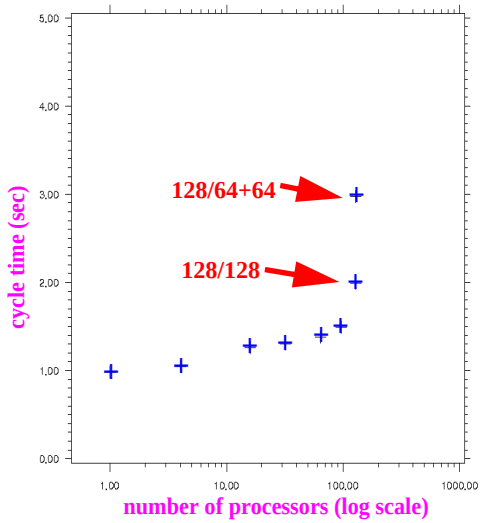


Figure 5. Cylinder problem performance on Blue Mountain.

The next performance results, shown in Figure 5., is for the ASCI Blue Mountain Technology Refresh platform, located at Los Alamos National Laboratory, New Mexico. Blue Mountain shows excellent linear scaling until 128 processors are in use. Two scaling numbers for 128 processors are shown. The first is using all 128 processors on one box (128/128), and the second using two boxes, 64 processors per box (128/64+64). Discussing the slowdown with other Blue Mountain users when all processors on the box were used yielded the following explanation: the Blue Mountain machine needs a few processors per box to handle message passing and other needs, and thus does not scale well to 128 processors per box. Furthermore, running across boxes has not been fully explored. Please note that this platform has the fastest processor speed of the ASCI machines.

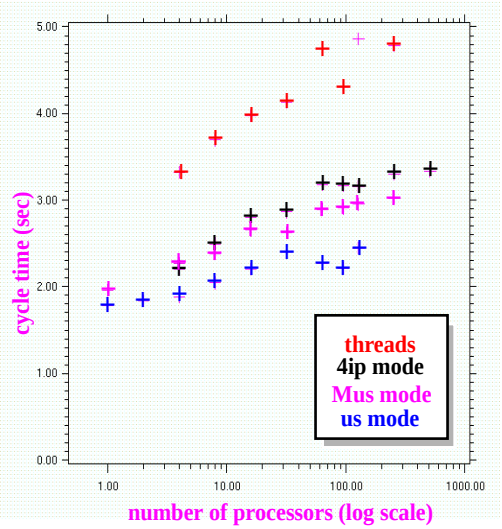


Figure 6. Cylinder problem performance on Blue Pacific.

The third performance measurement, shown in Figure 6., is the scaling study for the ASCI Blue Pacific Technology Refresh machine, located at Lawrence Livermore National Laboratory, California.
The Blue Pacific machine was tested using several communication modes:
- threads;
- us mode run with one processor per node;
- Mus mode “multiple us mode”, using four processors per node.
- 4ip mode run with four processors using IP protocol.

The best timings and best scaling results are with the us mode communication method. When the final delivery machines are in place, a version of Mus is slated to be the communication mode. The worst performance is with threads. However, the threading of ALE3D, and the correct use of compiler and processor settings has not been fully explored, and the hope is for better results then what is shown here.

Finally, in Figure 7., we put all three platform results together in one graph.

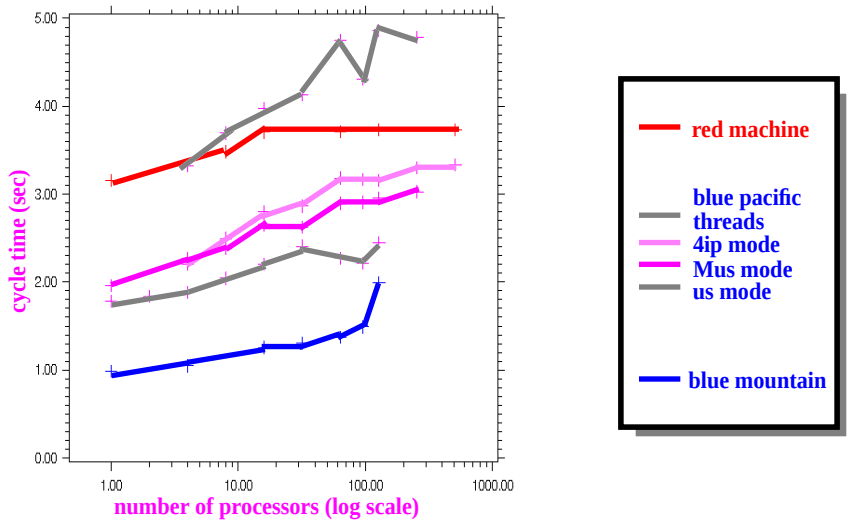


Figure 7. Cylinder problem performance on ASCI platforms.

The Sphere Problem. The Sphere problem, shown in Figure 8., incorporates slide surfaces and exhibits a more complex communication pattern. This problem has 6 materials, arranged in concentric shells. Interior materials are advecting, and the problem has three slide surfaces. There are 20,020 elements per domain, and three mesh refinements, fine, medium, and coarse zoned. The problem is decomposed using spectral methods. One interesting aspect of this problem is the resulting imbalance of the work load based on the decomposition method, pictured in Figure 9. In contrast to the Cylinder problem, communication patterns are not predictable.

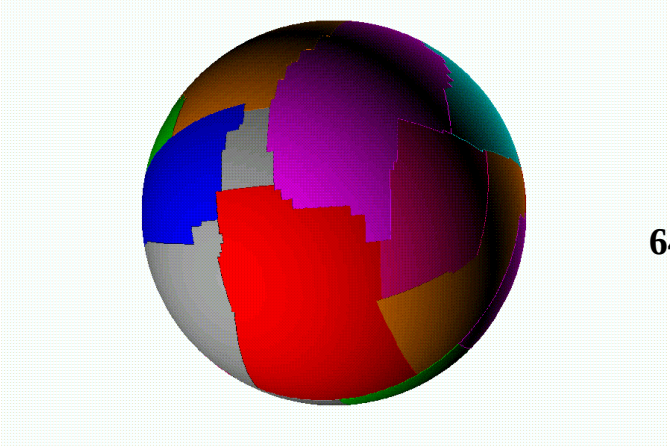


Figure 8. The Sphere Problem.

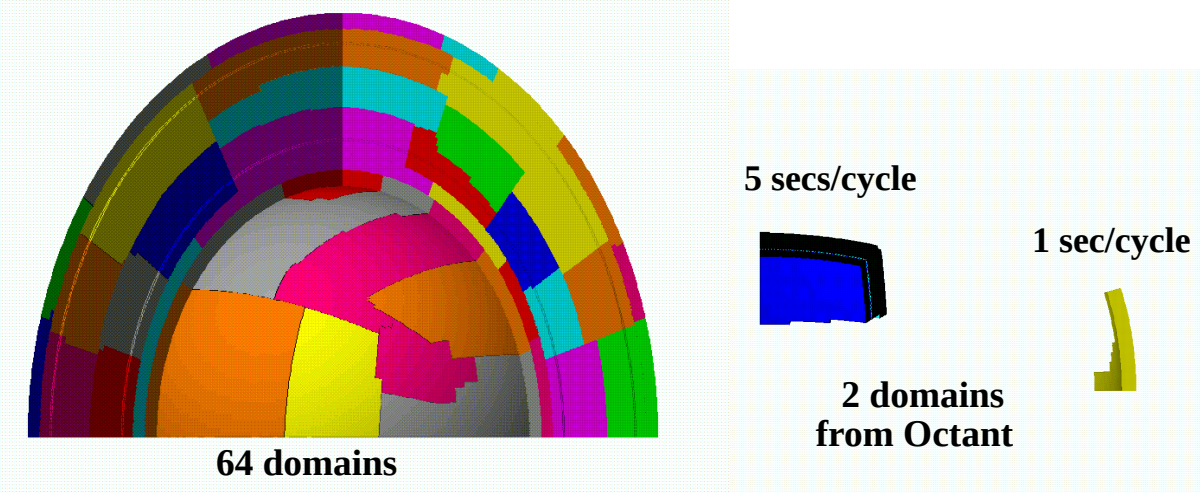


Figure 9. Load Imbalance with the Sphere Problem.

The first scaling numbers, seen in Figure 10., are for the Sphere problem without slide surfaces, so that we can show the additional complication that these surfaces add.

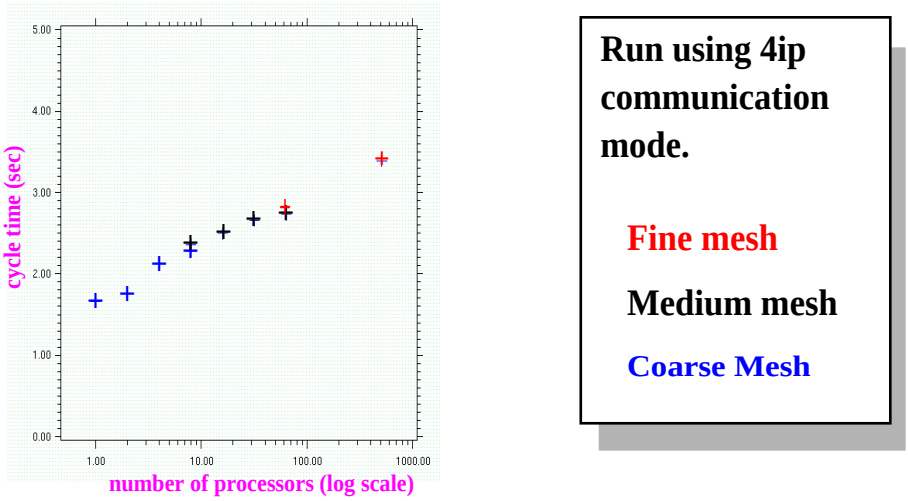


Figure 10. The Sphere problem's performance on Blue Pacific without slide surfaces.

The next performance numbers, shown in Figure 11., are the results of the scaling study of the Sphere with slide surfaces, run in several modes, using three different meshes. Several interesting

issues need to be discussed and resolved concerning these runs. Threading performance, and the jumps in timings for the large problems as they approach 128 processors needs studying.

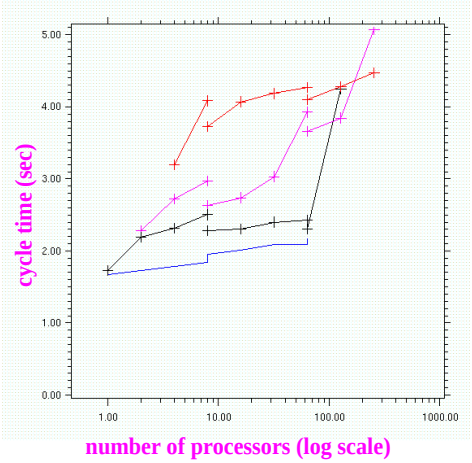


Figure 11. The Sphere problem's performance on Blue Pacific with slide surfaces.

Conclusions

ALE3D Hydro demonstrates portability and scalability on all three ASCI platforms. Additionally, each platform has unique characteristics that must be addressed for optimal performance. We need to use the dual processor on the Red machine. Threading performance on the Blue Pacific Technology Refresh platform needs to be investigated. Multi-box computing on the Blue Mountain Technology Refresh machine must be explored, as well as threading. Please note that Blue Pacific and Blue Mountain hardware and system software are evolving rapidly. This work was performed under the auspices of the U. S. Department of Energy by the Lawrence Livermore National Laboratory under Contract W-7405-ENG-48.