SAN099-0879J

ASCI Production Visualization Environments

Philip D. Heermann
Sandia National Laboratories

Abstract

The delivery of the first one tera-operations/sec computer has significantly impacted production data visualization, affecting data transfer, post processing, and rendering. Terascale computing has motivated a need to consider the entire data visualization system; improving a single algorithm is not sufficient. This paper presents a systems approach to decrease by a factor of four the time required to prepare large data sets for visualization. For daily production use, all stages in the processing pipeline from physics simulation code to pixels on a screen, must be balanced to yield good overall performance. Also, to complete the data path from screen to the analyst's eye, user display systems for individuals and teams are examined. Performance of the initial visualization system is compared with recent improvements. "Lessons learned" from the coordinated deployment of improved algorithms also are discussed, including the need for 64 bit addressing and a fully parallel data visualization pipeline.

Keywords: Accelerated Strategic Computing Initiative, Visualization, Isosurfacing, Systems Engineering, Massive Parallel Processing

1 Introduction

The delivery of the first one tera-ops/sec computer has significantly impacted data visualization. The one trillion operations per second machine, is the first high-speed machine from the U.S. Department of Energy's Accelerated Strategic Computing Initiative (ASCI) Program. The machine, also known as ASCI Red, is a massively-parallel distributed memory machine containing over 9000 Intel Pentium-Pro processors. The primary purpose of the ASCI Red machine is to greatly decrease the simulation time for large physics calculations

The leap forward in compute technology has impacted all aspects of visualizing simulation results. The data sets produced by this machine can greatly overwhelm common networks and storage systems. Data file formats, networks, processing software, and rendering software and hardware must be improved. A Systems Engineering approach is necessary to achieve improved performance. The common approach of improving a single algorithm can actually decrease performance of the overall system.

The user display environment is also important. Both individuals and teams require visualization environments to examine the detail in the large complex data sets. Common desktop hardware and software systems with 1024 X 1280 pixel displays (~1.3 Mpixels) are poorly suited for displaying data sets with 100's of million elements. Environments for teams require additional considerations to make effective use of the team. Projection display systems incorporating stereo displays and a capability to rapidly switch between a several image sources allow teams to perform efficiently.

To explore the issues of very large data sets, the discussion will first consider the workflow of an analyst and present a Systems Engineering method to improve the workflow process. The next section will present a systems perspective of information flow with an emphasis on team environments. The discussion will then focus on a case study example demonstrating an application of the systems approach to improve an early production visualization system. This initial system required over 16 hours to preprocess data for visualization. The goal of work presented here was to reduce this time to 4 hours to allow single day analysis of large data sets. This system performance goal required a balanced combination of hardware and software. The case study sections will present the visualization task, the initial system performance, and the solutions to improve process. The enhancement to display facilities will also be discussed.

2 Analyst Workflow

The discussion here considers systems that are used daily by analysts. In this production environment, entire system throughput is as important as the efficiency of a single system component. Many visualization algorithms achieve improved interaction or rendering performance at the expense of preprocessing the data. The preprocessing, however, can cost more in processing time or I/O than the benefit of the improved algorithm.

Consider the analysis loop in Figure 1. An engineer or physicist usually operates on a cycle with three basic stages, analysis preparation, physics computation, and analysis of results. During analysis preparation, problem topology, boundary conditions, physics models and other input to the simulation code are determined. This input is used by the second stage, the physics simulation, which generates output for analysis. Some smaller simulations are performed interactively, but many of the simulations requiring many hours or days are performed in a batch-processing mode.

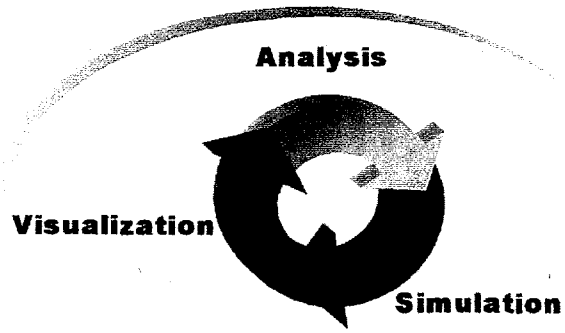

Figure 1: Computational Physics Analysis Loop

From a Systems Engineering perspective, minimizing the time to complete this analysis loop is the objective. Optimizing one stage at the expense of another stage may slow the loop cycle and increase the time required for analysis, thus lowering the productivity of the analyst. For maximum productivity, the preparation, simulation and data analysis stages must function together. From this perspective, it may be beneficial to slightly increase simulation time by writing to data visualization friendly format instead of selecting a fast output format that requires extensive post processing for visualization. The acceleration of the ASCI program exacerbates the need for system balance because a simulation code leveraged by the ASCI Red machine can completely overwhelm all supporting networks and computers. To illustrate this, consider the Sandia structured-grid shock physics code, CTH.

For CTH, 20-50 million cell calculations are routine. Three hundred million cell calculations can be easily run and a few one billion cell runs have been completed as proof of principle calculations. A 300 million-cell calculation generates approximately 50 gigabytes per time dump. Three days of running on the ASCI Red machine can produce 100 compressed dump data sets on the order of 350 gigabytes.

Data sets of this size quickly overwhelm most visualization systems. Simply transferring 350 gigabytes on a 100 megabit/sec Ethernet requires about 10 hrs. Even if the data is on a relatively fast 50 megabytes/sec disk RAID system, it would require 2 hours to stream the data into a post processing system, assuming that sufficient processing power is available to analyze or prepare the data for visualization at that data rate. In practice, most graphics tools are scalar and designed for much smaller data sets. On ASCI data sets, common commercial tools can take tens of minutes to hours to load and produce the first image. Much of this time is a result of the tool starting from raw data and producing isosurfaces, streamlines, or other features at initial startup. For faster loading and interaction, it is desirable to preprocess raw data into data sets that can be directly rendered.

3 Information Flow

Just as buildings are built layer upon layer, the flow of information can be considered as layers. This perspective can be stated in the simple acronym, "DIKJ," which stands for Data, Information, Knowledge, Judgement[1]. The simplest form of information is "Data" which is simply a collection or set of symbols. An example is raw numerical data from a test or experiment arranged in arbitrary order. The next level, "Information," is the arrangement of the data into a useful form such as a table or a graph. The following level is "Knowledge", this is an understanding of the table or graph. For instance, by examining a graph of decreasing velocity versus time for an automobile, it is possible to understand if the vehicle is rolling to a stop, braking or skidding. The final step is to take the understanding and use it to perform an action or make a judgement. For example combining a graph of brake pedal displacement with the previous deceleration curve, it is possible to determine that fully pressing the brake pedal while the car gently rolls to a stop means that the brakes are defective and the judgement is to get the vehicle repaired.

A common purpose for team meetings is to make decisions (i.e. make a judgement). In this setting, maximum efficiency for the team could be considered as maximizing the time spent making the judgement. Time spent distilling data into information and knowledge for presentation to the team is simply a waste of the other team members' time. In fact, having all the data refined to knowledge and information is common. Normally, team meetings have a presentation of material, which is simply a distillation of the data into the few key pieces of information or knowledge that are important to the judgment.

Considering this process in the ASCI context is revealing. ASCI physics codes commonly output raw data, which is post-processed by visualization and analysis packages. In the team setting, this refinement of the data is often done

before the meeting and viewgraphs or time sequence animations are presented. This preprocessing, however, has drawbacks. Often in a team discussion there is a desire to consider a different variable or view a certain area in more detail, which is not possible in common conference room environments. This can delay the judgement by requiring the presenter to do more analysis and convene another meeting or the team may make the judgement based on the available data plus a priori knowledge from experienced team members.

The more desired condition is illustrated in Figure 2. Here the "DIK" layers are automated as much as possible to allow real time requests for new information to be derived. Raw data from a broad range of sources, such as test data, physics simulation data, or manufacturing data, can be quickly accessed and refined into knowledge. The teams' focus is spent on the decision making with the full benefit of quick resolution of unforeseen questions. Thus interdisciplinary teams can readily and quickly consider a broad range of issues with an ultimate goal of rapid high quality judgement. This capability also works to shorten the design cycle. High quality decisions lead to less rework late in the design cycle and an ability to make decisions more rapidly directly shortens development time.

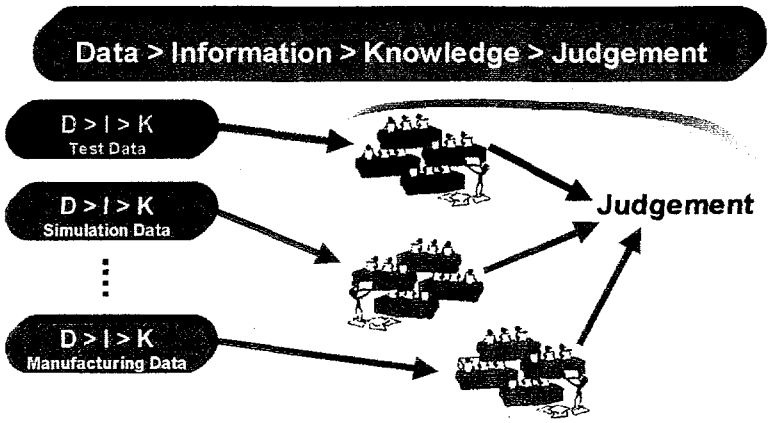

Figure 2: Information Flow and Automation of Information Flow to enable Team Judgement.

4 Visualization Task

The visualization task considered here is isosurface visualization of shock physics data generated by Sandia's CTH code. The data is a structured grid where matter moves through a 3-D grid in time. Each cell in the grid can contain empty space or one, two or as many as 20 materials. This is recorded in each cell as a "volume fraction," where one indicates the cell is completely full of a material and zero indicates no material in the cell. For multiple materials the sum of the volume fractions plus the "void fraction" equals one. For visualization, 3-D isocontours are generated from the volume fractions for each material and the resulting polygonal surfaces are rendered.

To rapidly explore the database, the rendering and isosurface generations were performed as separate tasks. The processing consisted of the following stages:

1. Simulation run on MP machine,
2. Transfer data to visualization server,
3. Isosurface extraction[2] and polygon decimation[3],
4. Concatenation of polygon files,
5. Real-time exploration of isosurfaces.

The simulation run on the MP machine produces one file for each processing node. Thus, a simulation on 2500 nodes produces 2500 files, each containing that node's portion of the overall problem. The 2500 files are transferred across a network to a large visualization machine for the remainder of the steps. On the visualization server, each file is processed independently to extract the isosurfaces contained in each file. The list of polygons is decimated in memory prior to writing the polygons to disk. The resulting 2500 polygon files for each time dump are concatenated into a single file for easy loading by the rendering software. This isosurface/decimation/concatenation process is repeated for each of 10-100 time dumps contained in the original node files. The resulting concatenated polygon files, one for each time step, are selectively loaded into memory for interactive rendering with Sandia's Eigen/VR[4]. This process provides interactive exploration of the data set limited only by the rendering performance of the graphics systems.

5 Initial System Performance

The initial visualization system was developed for visualizing results from an 1860 node Intel Paragon MP supercomputer. The network system was a 10-megabit Ethernet between the Paragon and a Silicon Graphics ONYX for visualization. The ONYX system was configured with four R4400 250 MHz processors, two REALITY II graphics pipes, 6 gigabytes of main memory, and 96 gigabytes of RAID 0 disk. This system was capable of the following processing times for a 100 million-cell calculation computed on the ASCI Red machine:

Table 1: Initial Visualization System Performance

Task	Processing Time	Notes
Simulation Time	17 hours	6 Gbyte database
Data Transfer	5.5 hrs	10 Mb Ethernet
Isosurfacing/ Decimation	11 hours	20-30 min./timestep
Concatenation	~ 1 minute	Disk Bandwidth limited
Frame Rendering Time	~104 sec/frame	~5% of peak graphics pipe efficiency
Total Postprocessing Time	16.5 hours	

The rendering times shown in Table 1 are a single decimated frame containing 13.7 million triangles. All stages in the processing path were considered to be inadequate for TeraFLOPS problems. A goal was set to reduce the total processing time to less than 4 hours. This goal would permit 100 million cell calculation to prepared for exploration in a morning of work.

Also, the rendering system had several issues that needed to be resolved. First, Eigen/VR was a 32-bit code, which was limited to a 2-gigabyte memory image (i.e., only 1-2 time-steps in memory). The rendering speed was too slow at ~104 seconds/frame for easy interaction. Many other visualization tools, which integrate the isosurface generation with rendering, were tested, but all required minutes to tens of minutes to generate frames.

The initial user interface equipment consisted of two graphics displays (1024 X 1280 pixels), a Fly Box™ joystick controller, and a FakeSpace™ Boom. The biggest issue was the slow rendering speed, the Boom was difficult to use because moving to a new position and waiting more than a minute for an update was impractical. The display resolution was not as much of a problem, but the zooming in and out of the image was a frequent operation to examine areas in more closely. The joystick, however, remained useful providing a third degree of freedom (twist), which was usefully mapped to raising and lowering the virtual eye location.

6 Improved Production System

To meet the 4-hour time and improved interaction goals, the entire post-processing path from the physics simulation code to the analyst's interaction with the visualization system was improved. The visualization server was improved to a Silicon Graphics ONYX II visualization system with sixteen 190 MHz R10000 processors, four INFINITE REALTY graphics pipes, 32 Gigabytes of memory, and 1.5 terabytes of Fiber Channel connected disk. The network was improved from a single Ethernet to four ATM OC/3 channels increasing network bandwidth to 155 Megabits/sec per channel.

To improve the software, several research and development paths were explored. The isosurface software, a marching-cubes algorithm, was parallelized [5] and the concatenation function was integrated into the software. The rendering software was rewritten to accommodate a 64-bit memory image and improve the efficiency of using the graphics hardware. Although the software is undergoing continuous evolution, the current enhancements for the 100 million-cell CTH data set are presented in Table 2.

Table 2: Current Data Visualization System Performance

Task	Processing Time	Notes
Simulation Time	17 hours	6 Gbyte database
Data Transfer	5 minutes	24 Mbytes/sec
Isosurfacing/ Decimation	3.7 X speedup on 4 processors	Exclusive of I/O. Parallel file I/O is still under development
Concatenation	Integrated with Isosurfacing	
Frame Rendering Time	~8.2 seconds/frame	~15% peak graphics pipe utilization
Approx. Total Postprocessing Time	3.5 hours	

The parallelization of the isosurface/decimation algorithms shifted the bounding limit from processor limited to I/O limited. Thus, current work is focused on improving I/O performance [6]. Also improved rendering rates are still desired. The graphics pipe efficiency increase was due to many small improvements; however, the major gain in performance was achieved by replacing polygon lists with triangle strips. The triangle stripe conversion adds a 10% overhead to the isosurfacing time The current 12 fold improvement in rendering speed is due to approximately a four fold increase in hardware speed and an approximately three fold increase in software efficiency.

The improved display environment used the next generation monitors with 1280 X 1920 pixels, and a Logitech™ Magellen 4DOF input device. The accelerated rendering was the biggest gain for the user display environment, but the Magellen device enabled a more intuitive control for navigation. No formal study was conducted, but the increased display resolution seems to have little impact on the overall display environment.

We applied significant resources to improving the team working environments. Several conference rooms were equipped with display systems including a simple stereo projection system, a 2X2 POWERWALL[7], and two high end visionarium systems with three projectors on a curved screen. All the rooms have the ability to quickly switch between multiple sources of images including UNIX™ and NT™ computers, VCR's and DVD players. One of the visionariums is shown in Figure 3. The visionariums are designed to explore the concepts that were presented in Section 3.

Figure 3: Sandia National Laboratories – Visualization Design Center, Livermore, CA.

7 Lessons Learned

The improved hardware and software have greatly decreased the pipeline latency for isosurface exploration of structured data sets. The speedup was accomplished by improving all stages of the pipeline. During the process, several lessons were uncovered that apply to all stages in the pipeline. The four primary lessons learned during this work are:

1. buffer and 32bit addressing limits,
2. stay parallel at all points in the path,
3. balance all components, and
4. the importances of parallel I/O.

Primary issues impacting the ASCI visualization are buffer and memory limits. These problem surfaces in two main problem areas, insufficient buffer sizes and 32bit limited addressing. Many tools, like the Unix csh, have internal buffers that overflow when presented with 2500 files in a directory. A prime example is "cp *.dat." This is a common way to transfer files on a disk system from one directory to another location. The csh, while generating the list of file names for the "cp" command, overflows a buffer and issues a copy with less than the full number of files. Recursive copy, tar, and other tricks provide a work-around to problems but the issue is that Unix on many machines may have 64 bit file systems and still have difficulty with large numbers of files.

The second issue, 32 bit addressing, is simply that files and memories larger than 2 gigabytes require 64 bit addressing. A classic example of this is the rendering code. Managing polygon lists for multiple variables across multiple time dumps easily exceed 2 gigabytes. Initially Sandia's Eigen/VR code was used for rendering, however, it was coupled to a variety of virtual reality input devices. Many of these had only 32 bit driver libraries. Therefore, it proved to be faster rewriting the rendering software rather than disentangle all the 32 bit dependencies.

It is also very important to stay parallel at all stages in the pipeline. Any time the processing drops to a single processor it proves to be a bottleneck. A classic example is that in the quest to speed rendering, it was decided to switch from polygon lists to triangle strips. This introduced another step into the pipeline. The initial polygon list to triangle strip conversion was a uniprocessor application (i.e., adds hours to the preprocessing path). After tuning and parallelization, however, the triangle strip conversion was reduced to approximately 4 minutes/per frame (~14 million triangles frame). Further improvements to the triangle strip code, however, will require integration into the output portion of the isosurface/decimation code. The integration is necessary to eliminate writing intermediate results to the file system. Any file I/O is time consuming when working with tens of gigabytes.

The need to couple the isosurface/decimation code to the triangle striping code is an example of the third issue. The requirement to couple the codes is generated by attention to balancing the overall system. The simplest way to deploy the triangle-striping code would be to simply read the output files of isosurface/decimation code and convert them to triangle-strips. This initial solution, however, adds two file system accesses to the processing pipeline, which greatly delays the processing path.

The final issue is parallel I/O. The need for good parallel I/O can not be understated. On the current visualization server with 16 processors the disks sustain read rates above 300 megabytes/sec. Dividing the disk bandwidth by the number of processors yield slightly less than 20 megabytes/sec per processor. This is a mediocre bandwidth considering the speed of existing processors. Sixteen processors all trying to access disk at the same time, however, delivers miserable performance as 16 requests fight for the disk heads. To mitigate these problems, file formats and libraries need to consider support for coordinated I/O. Both parallel data transfers and parallel processing tools are critical for rapid visualization of ASCI data sets.

The visionarium facilities were completed only in the last several months. Teams have begun using the facilities, however, the full capabilities have yet to be realized. A significant issue is the operating the environment. Although the controls are not difficult, the flexibility of the rooms can be intimidating to new users. Currently, we have found the best use of the facilities by adding skilled users of the software tools to the team. This allows the decision/ judgement team to focus on the issues at hand while the skilled software users manage the locating, loading and displaying of the information. In a sense the rooms are used like NASA's Mission Control Room or ship control room where skilled operators provide information to others. The primary difference is that the spectrum of activity is less tightly defined than is common in other settings. Also, The rooms have highlighted the need for high performance flexible networks. Having a file on your desktop computer that is inaccessible or downloading data for tens of minutes is certainly counter productive to the goal of the visionarium.

8 Acknowledgements

The work to improve the production visualization system depends on many researchers. Pat Crossno and Dino Pavlakos have worked to parallelize the isosurface and decimation code. Marlin Kipp provided the CTH data sets. The Sandia networking staff and Marty Barnaby have greatly improved the network hardware and software systems. Pang Chen has worked to improve the parallel file systems. Jake Jones authored the 64 bit rendering code. Visualization facility development was work of many talented individuals, including Jerry Friesen, Jeff Jortner, Carl Leishman and Dan Zimmerer. Without the talents and time of these researchers and many others this work would not be possible. This work is supported by United States Department of Energy under contract DE-AC04-94AL85000. Additional funding provided by Sandia National Laboratories internal research program.

7 References

[1] Koen, Billy V., University of Texas at Austin, Personal Communication, 1990.

[2] Lorensen, W.E. and H.E. Cline, "Marching Cubes: A High Resolution 3D Surface Construction Algorithm," in Computer Graphics (SIGGRAPH '87 Proceedings). 1987. pp. 163-169.

[3] Schroeder, W.J., J.A. Zarge, and W.E. Lorensen, "Decimation of Triangle Meshes," Computer Graphics (SIGGRAPH '92 Proceedings), 1992. 26(2): pp. 65-70.

[4] http://www.cs.sandia.gov/SEL

[5] Crossno, P.J., D.D. Cline, and J.N. Jortner, "A Heterogenous Graphics Procedure for Visualization of Massively Parallel Solutions," FED-Vol 156, CFD Algorithms and Applications for Parallel Processors, ASME Fluids Engineering Conference, Washington D.C., June 20-24, 1993, pp. 65-70

[6] Sturtevant J. E., M.A. Christon, P.D. Heermann, P. Chen: PDS/PIO: Lightweight Libraries for Collective Parallel I/O, accepted for publication in the Proceedings of SC98, Orlando, FL, Nov 7-13, 1998

[7] Woodward, Paul, University of Minnesota, Personal Communication, 1993.