A GPU-Enhanced Cluster for Accelerated FMS

Dan M. Davis, Robert F. Lucas, Gene Wagenbreth, John J. Tran and James R. Moore
Information Sciences Institute, University of Southern California, Marina del Rey, California
{ddavis, rflucas, genew, jtran, & jjmoore} @isi.edu

Introduction This paper addresses the experience and the experiments of the authors with the new GPU accelerator-enhanced Linux Cluster at JFCOM's Experimentation Directorate, J9. Requirements, design considerations, configuration decisions, and early experimental results are reported. The J9 simulations need to be done at a scale and level of resolution adequate for modeling the complexities of urban combat. The J9 code is the current instantiation of a long lineage of entity-level battlefield codes such as JSAF. To power these activities, JFCOM requires an enhanced Linux cluster of adequate size, power, and configuration to support simulations of more than 2,000,000 entities on global-scale terrain.

Objective The objective of this research is to provide 24x7x365, enhanced, distributed and scalable compute resources to support joint warfighters at JFCOM as well as the U.S. military services and international defense partners. This enables them to develop, explore, test, and validate 21st century battlespace concepts in J9's JFL. The specific goal is to enhance global-scale, computer-generated experimentation by sustaining more than 2,000,000 entities on appropriate terrain, along with valid phenomenology. The authors sought to achieve this using the Graphics Processing Units (GPUs) as general-purpose accelerators in a cluster architecture.

Methodology The method employed was to use existing DoD simulation codes on the advanced Linux clusters operated by JFCOM. The improved cluster reported herein supplants the current JFCOM J9 DC clusters with new upgraded 64-bit CPUs and enhanced with nVidia 8800 GPUs. Further, the authors have begun to modify legacy codes to make optimal use of the GPUs' substantial processing power. Initially, the major driver for the FMS community's use of accelerator-enhanced nodes was the need for faster processing to accomplish line-of-sight calculations. However, the first experiments were conducted on a smaller code set, one also amenable to GPU acceleration, to facilitate the programming and hasten the experimentation insights.

Results The learning curve for the use of the new C-like CUDA code for GPU non-graphics processing was found to be manageable. It was demonstrated that the GPU could be very effective at reducing the time spent factoring the large frontal matrices near the root of the elimination tree in the strategic calculation approach. The GPU accelerated the overall factorization at close to the factor of two originally hypothesized. Results from the GPU-enhanced cluster itself should be forthcoming soon.

A GPU-Enhanced Cluster for Accelerated FMS

Dan M. Davis, Robert F. Lucas, Gene Wagenbreth, John J. Tran and James R. Moore
Information Sciences Institute, University of Southern California, Marina del Rey, California
{ddavis, rflucas, genew, jtran, & jjmoore} @isi.edu

Abstract

The Forces Modeling and Simulation (FMS) community has often been hampered by constraints in computing: not enough resolution, not enough entities, not enough behavioral variants. High Performance Computing (HPC) can ameliorate those constraints. The use of Linux Clusters is one path to higher performance; the use of Graphics Processing Units (GPU) as accelerators is another. Merging the two paths holds even more promise. The High Performance Computing Modernization Program (HPCMP) accepted a successful proposal for a new 512 CPU (1024 core), GPU-enhanced Linux Cluster for the Joint Forces Command's Joint Experimentation Directorate (J9). The basic concepts underlying the use of GPUs as accelerators for intelligent agent, entity-level simulations are laid out. The simulation needs of J9, the direction from HPCMP and the careful analysis of the intersection of these are explicitly discussed. The configuration of the cluster is addressed and the assumptions that led to the conclusion that GPUs could increase performance by a factor of two are offered. Issues, problems and solutions will all be reported objectively, as guides to the FMS community and as confirmation or rejection of early assumptions. Early characterization runs of a single CPU with GPU-enhanced extensions are reported.

1. Introduction

This paper addresses the experience and the experiments of the authors with the new GPU accelerator-enhanced Linux Cluster at JFCOM. Requirements, design considerations, configuration decisions, and early experimental results are reported.

1.1. Joint Forces Command Mission and Requirements

The mission of the Joint Forces Command (JFCOM) is to lead the transformation of the United States Armed Forces and to enable the U.S. to exert broad-spectrum dominance as described in Joint Vision 2010 (CJCS, 1996) and 2020 (CJCS, 2000). JFCOM's research arm is the Joint Experimentation Directorate, J9. This leads to the virtually unique situation of having a research activity lodged within an operation command, which then calls for experiments in which warfighters in uniform are staffing the consoles during interactive, HPC-supported simulations.

J9 has conducted a series of experiments to model and simulate the complexities of urban warfare using well-validated entity-level simulations, *e.g.* Joint Semi-Automated Forces (JSAF) and the Simulation of the Location and Attack of Mobile Enemy Missiles (SLAMEM). These need to be run at a scale and resolution adequate for modeling the complexities of urban combat.

The J9 code is the current instantiation of a long lineage of entity-level battlefield codes. It consists of representations of terrain that are populated with intelligent-agent friendly forces, enemy forces and civilian groups. All of these have compute requirements to produce their behaviors. In addition, a major computational load is imposed in the performance of line-of-sight calculations between the entities. This is a problem of some moment, especially in the light of its

inherently onerous "n-squared" growth characteristics (Brunett, 1998). Consider the case of several thousand entities needing to interact with each other in an urban setting with vegetation and buildings obscuring the lines of sight. This situation has been successfully attacked by the use of an innovative interest-managed communication's architecture (Barrett, 2004).

To power these activities, JFCOM requires an enhanced Linux cluster of adequate size, power, and configuration to support simulations of more than 2,000,000 entities within high-resolution insets on a global-scale terrain database. This facility will be used occasionally to interact with live exercises, but more often will be engaged interactively with users and experimenters while presenting virtual or constructive simulations. (Ceranowicz, 2005) It must be robust to reliably support hundreds of personnel committed to the experiments and it must be scalable to easily handle small activities and large, global-scale experiments with hundreds of live participants, many distributed trans-continentally, as shown in Figure 1 below.

1.2. Joint Futures Lab (JFL)

The goal of the JFL is to create a standing experimentation environment that can respond immediately to DoD time-critical needs for analysis. It is operating in a distributed fashion over the Defense Research and Engineering Network (DREN), at a scale and level of resolution that allows JFCOM and its partners to conduct experimentation on issues of concern to combatant commanders, who participate in the experiments themselves. The JFL consists of extensive simulation federations, software, and networks joined into one common computer infrastructure that is supporting experiments. This standing capability includes quantitative and qualitative analysis, flexible plug-and-play standards, and the opportunity for diverse organizations to participate in experiments.

1.3. Joint Advanced Training and Tactics Laboratory (JATTL)

JATTL was established to support mission rehearsal, training, operational testing, and analysis. The principle concerns of the JATTL are developing technologies that support the pre-computed products required for joint training and mission rehearsal. This is being explored under the Joint Rapid Distributed Database Development Capability as well as others required during its execution. The latter include phenomenology such as environment, cultural assets, civilian populations, and other effects necessary to represent real operations. The JATTL is connected via both DREN and the National Lambda Rail (NLR) to over thirty Joint National Training Capability sites nationally.

1.4. JFCOM's JESPP

A J9 team has designed and developed a scalable simulation code that has been shown capable of modeling more than 1,000,000 entities. This effort is known as the Joint Experimentation on Scalable Parallel Processors (JESPP) project (Lucas, 2003.) This work builds on an earlier DARPA/HPCMP project named SF Express. (Messina, 1997)

The early JESPP experiments on the University of Southern California Linux cluster (now more than 2,000 processors) showed that the code was scalable well beyond the 1,000,000 entities actually simulated, given the availability of enough nodes (Wagenbreth, 2005).

The current code has been successfully fielded and operated using JFCOM's HPCMP-provided compute assets hosted at ASC-MSRC, Wright Patterson AFB, and at the Maui High Performance Computing Center (MHPCC) in Hawai'i. The J9 team has been able to make the

system suitable and reliable for day-to-day use, both unclassified and classified.

This effort is needed in order to deliver a state-of-the-art capability to military experimenters so they can use it to easily initiate, control, modify, and comprehend any size of a battlefield experiment. It now additionally allows for the easy identification, collection, and analysis of the voluminous data from these experiments, enabled by the work of Dr. Ke-Thia Yao and his team (Yao, 2005).

A typical experiment would find the JFCOM personnel in Suffolk Virginia interfacing with a "Red Team" in Fort Belvoir Virginia, a civilian control group at SPAWAR San Diego, California, participants at Fort Knox Kentucky and Fort Leavenworth Kansas, all supported by the clusters on Maui and in Ohio. The use of interest-managed routers on the network has been successful in reducing inter-site traffic to low levels.

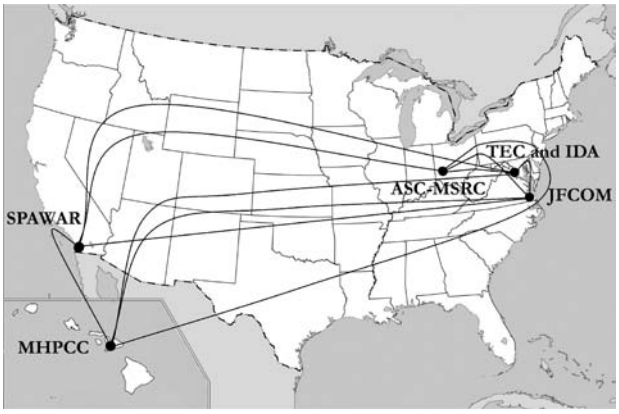

Figure 1

JFCOM's Experimentation network: MHPCC, SPAWAR, ASC-MSRC, TEC/IDA and JFCOM

Even using these powerful computers, the JFCOM experimenters were constrained in a number of dimensions, *e.g.* number of entities, sophistication of behaviors, realism of various environmental phenomenology, *etc.* While the scalability of the code would have made the use of larger clusters feasible, a more effective, efficient, economical and elegant solution was sought.

1.5. Broader Impacts for the HPCMP Community

The discipline of Forces Modeling and Simulation (FMS) is unique to the DoD, compared to many of the other standard science disciplines, *e.g.* CFD and Weather. In a similar way, interactive computing is a new frontier being explored by the JESPP segment of FMS, coordinating with a few other user groups. Along these lines, the newly enhanced Linux Cluster capability will provide significant synergistic possibilities with other computational areas such as signals processing, visualization, advanced numerical analysis techniques, weather modeling and other disciplines or computational sciences such as SIP, CFD, and CSM.

2. Objective

The objective of this research is to provide 24x7x365 enhanced, distributed and scalable compute resources to enable joint warfighters at JFCOM as well as its partners, both U.S. Military Services and International Allies. This enables them to develop, explore, test, and validate 21st century battlespace concepts in JFCOM J9's JFL. The specific goal is to enhance global-scale, computer-generated support for experimentation by sustaining more than 2,000,000 entities on appropriate terrain, along with valid phenomenology.

The quest to explore broader use of GPUs is often called GPGPU, which stands for General Purpose computation on GPUs (Lastra 2004). While the programming of GPUs has been pursued for some time, the newly released Compute Unified Device Architecture (CUDA) programming language (Buck, 2007) has made that effort more

accessible to experienced C programmers. For that reason, the HPCMP accepted the desirability of upgrading the original cluster configuration from nVidia 7950s to nVidia 8800, specifically to enable the use of CUDA. This met with HPCMP's goal of providing an operationally sound platforms rather than an experimental configuration that would not be utilized easily by the wider DoD HPC community.

3. Methodology

The method employed was to use existing DOD simulation codes on advanced Linux clusters operated by JFCOM. The effort reported herein supplants the current JFCOM J9 DC clusters with a new cluster enhanced with 64-bit CPUs and nVidia 8800 graphics processing units (GPUs). Further, the authors have begun to modify a few legacy codes.

As noted above, the initial driver for the FMS use of accelerator-enhanced nodes was principally the faster processing of line-of-sight calculations. Envisioning other acceleration targets is easy: physics-based phenomenology, CFD plume dispersion, computational atmospheric chemistry, data analysis, *etc.*

The first experiments were conducted on a smaller code set, to facilitate the programming and accelerate the experimentation. An arithmetic kernel from an MCAE "crash code" (Diniz, 2004) was used as vehicle for a basic "toy" problem. This early assessment of GPU acceleration focused on a subset of the large space of numerical algorithms, factoring large sparse symmetric indefinite matrices. Such problems often arise in Mechanical Computer Aided Engineering (MCAE) applications. It made use of the SGEMM (Single precision GEneral Matrix Multiply) algorithm (Whaley, 1998) from the BLAS (Basic Linear Algebra Subprograms) routines (Dongarra, 1993).

The GPU is a very attractive candidate as an accelerator for computational hurdles such as sparse matrix factorization. Previous generations of accelerators, such as those designed by Floating Point Systems (Charlesworth 1986) were for the relatively small market of scientific and engineering applications. Contrast this with GPUs that are designed to improve the end-user experience in mass-market arenas such as gaming.

The Sony, Toshiba, and IBM's (STI) Cell processors (Pham, 2006) are also representative of a new generation of devices whose market share is growing rapidly, independently of science and engineering. The extremely high peak floating point performance of these new commodity components encourages the consideration of ways in which they can be exploited to increase the throughput and reduce the cost of applications such as FMS, which are beyond the markets for which they were originally targeted.

In order to get meaningful speed-up using the GPU, it was determined that the data transfer and interaction between the host and the GPU had to be reduced to an acceptable minimum. For factoring individual frontal matrices on the GPU the following strategy was adopted:

1) Downloaded the factor panel of a frontal matrix to the GPU.
2) Stored symmetric data in a square matrix, not a compressed triangular.
3) Used a left-looking factorization, proceeding from left to right:
   a) Updated a panel with SGEMM
   b) Factored the diagonal panel block
   c) Eliminated the off-diagonal entries
4) Updated the Schur complement of this frontal matrix with SGEMM
5) Returned the entire frontal matrix to the host, converting back from square to triangular storage

6) Returned an error if the pivot threshold was exceeded or a diagonal entry was equal to zero

4. Results

The sum of the time spent at each level of a typical elimination tree is shown in Figure 2. The sum from all of the supernodes at each level is plotted as the red curve. The time spent assembling frontal matrices and stacking their Schur complements is represented by the yellow curve. These are the overheads associated with using the multifrontal method. The total time spent at each level of the tree when running on the host appears below in blue. The time spent factoring the frontal matrices is the difference between the blue and yellow curves. The time spent at each level of the elimination tree when using the GPU to factor the frontal matrices is brown in the graph below. The difference between the brown curve and the yellow one is the time spent on the GPU.

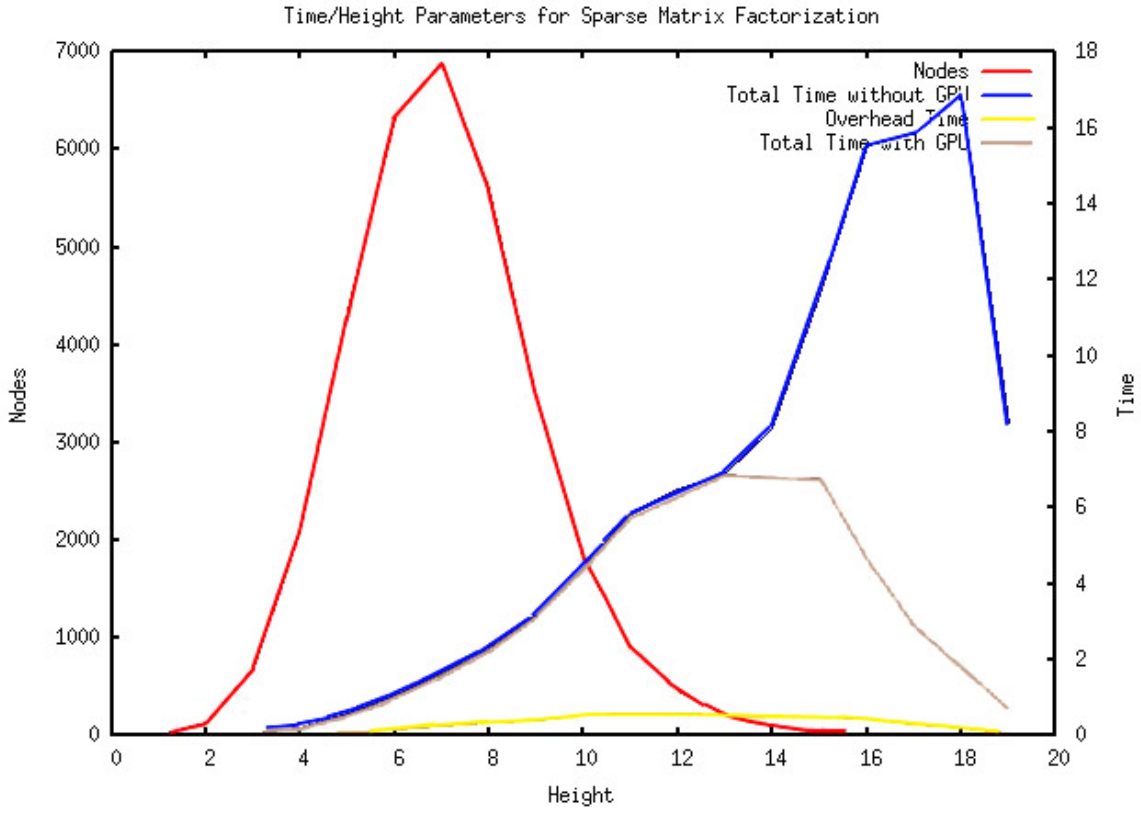

Figure 2

Number of Supernodes and time spent factoring each level of the elimination tree.

Looking at Figure 2, it seems apparent that the GPU is very effective at reducing the time spent factoring the large frontal matrices near the root of the elimination tree. The difference between the brown and blue curves is the cumulative time of 52.5 seconds by which the GPU accelerated the overall factorization. Similar results could be expected from similar codes.

The SGEMM function used in this work was supplied by nVidia. In testing, it was found that it could achieve close to 100 GFLOP/s, over 50% of the peak performance of the nVidia GTS GPU. Thus the efforts were focused on optimizing the functions for eliminating off-diagonal panels (GPUl) and factoring diagonal blocks (GPUd). A more detailed description of this technique can be found in an unpublished paper (Lucas, 2007).

5. Significance to DoD

This research will provide warfighters with the new capability to use Linux clusters in a way that will simulate the required throngs of entities and suitably global terrain necessary to represent the complex urban battlefield of the 21st Century. It will enable experimenters to simulate the full range of forces and civilians, all interacting in future urban battlespaces. The use of GPUs as acceleration devices in distributed cluster environments shows apparent promise in any number of fields. Further experimentation should extend the applicability of these concepts. The CUDA code proved to be easily exploited by experienced C programmers.

Another area of interest is the combination of GPUs and others accelerators such as FPGAs through the application of heterogeneous computing techniques. The successful use of FPGAs as accelerators has been reported (Lindermann, 2005) and there are facilities where FPGAs are installed on compute nodes of Linux clusters. Ideally, some future CPU-GPU-FPGA configuration would allow a designer to take advantage of the strengths of FPGA and GPU accelerators while minimizing their weaknesses. By combining a GPU and FPGA in this manner the raw integer power and reconfigurability of an FPGA is key to applications such as cryptography and fast folding algorithms (Frigo, 203). This could be added to the specialized computational power of GPUs, which can be optimized for operations such as those used in linear algebra (Fatahalian, 2004) and FFT operations (Sumanaweera, 2005).

Acknowledgements

Thanks are due to the excellent staffs at ASC-MSRC and MHPCC. While singling out any individuals is fraught with risks of omission, we could not let this opportunity pass without mentioning Gene Bal, Steve Wourms, Jeff Graham and Mike McCraney and thanking them for their stalwart and even heroic support of this project. Of course, the authors are grateful for the unstinting support of the nVidia staff in this early foray into CUDA and GPU use, most especially Dr. Ian Buck and Norbert Juffa. Some of this material is based on research sponsored by the Air Force Research Laboratory under agreement number FA8750-05-2-0204. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory or the U.S. Government.

References

Barrett, B. & Gottschalk, T.D., (2004), Advanced Message Routing for Scalable Distributed Simulations, I/ITSEC Conference, Orlando, FL

Brunett, S., & Gottschalk, T.D., (1998), A Large-scale Meta-computing Framework for the ModSAF Real-time Simulation, Parallel Computing:, V24:1873-1900, Amsterdam

Buck, I., (2007), GPU Computing: Programming a Massively Parallel Processor, International Symposium on Code Generation and Optimization, San José, California

Ceranowicz, A. & Torpey, M., (2005), Adapting to Urban Warfare, Journal of Defense Modeling and Simulation, 2:1, January 2005, San Diego, Ca

Charlesworth, A., & Gustafson, J., (1986), Introducing Replicated VLSI to Supercomputing: the FPS-164/MAX Scientific Computer, in IEEE Computer, 19:3, pp 10-23, March 1986

CJCS, (2000), Joint Vision 2020, Director for Strategic Plans and Policy, J5: Strategy Di-vision, Washington, D.C.: Government Printing Office

CJWC, (1997), Concept for Future Joint Operations, Commander, Joint Warfighting Center, Fort Monroe, VA.

Diniz, P., Lee, Y.-J., Hall, M. & Lucas, R., (2004), A Case Study Using Empirical Optimization for a large, Engineering Application, in the Proceedings of the 18th International Parallel and Distributed Processing Symposium, April 2004, pp: 200-208

Dongarra, J., (1993), Linear algebra libraries for high-performance computers: a personal perspective, Parallel & Distributed Technology: Systems & Applications, IEEE, Feb. 1993, Volume: 1, Issue: 1, pp: 17 - 24

Fatahalian, K., Sugerman, .J. & Hanrahan, P., (2004), Understanding the efficiency of GPU algorithms for matrix-matrix multiplication, Workshop on Graphics Hardware, Eurographics/SIGGRAPH

Frigo., J., Palmer, D., Gokhale, M., and M. Popkin-Paine, M. (2003), Gamma-ray pulsar detection using reconfigurable computing hardware, 11th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM),.

Lastra, A., Lastra, M. Lin, and D. Minocha, (2004), ACM Workshop on General Purpose Computations on Graphics Processors.

Linderman, R. W., Linderman, M. H., and Lin, C-S., (2005), FPGA Acceleration of Information Management Services, 2005 MAPLD International Conference, Washington, DC

Lucas, R.F., Wagenbreth, G., Tran, J.J., & Davis, D. M., (2007), Multifrontal Computations on GPUs, unpublished ISI White Paper, on line at: www.isi.edu/~ddavis/JESPP/2007_Papers/SC07/mf2_gpu_v0.19a-nms.doc

Lucas, R., & Davis, D., (2003), Joint Experimentation on Scalable Parallel Processors, 2003 I/ITSEC Conference, Orlando, FL

Joint Pub 1-02, (2000), Department of Defense Dictionary of Military and Associated Terms, Chairman of the Joint Chiefs of Staff, Washington, D.C.

Messina, P. C., Brunett, S., Davis, D. M., Gottschalk, T. D., (1997) Distributed Interactive Simulation for Synthetic Forces, In Mapping and Scheduling Systems, International Parallel Processing Symposium, Geneva

Pham, D. C., *et al.*, (2006), Overview of the Architecture, Circuit Design, and Physical Implementation of a First-Generation Cell Processor, IEEE Journal of Solid State Circuits, Vol. 41, No. 1, January, 2006

Sumanaweera, T. and Liu D., (2005), Medical Image Reconstruction with the FFT, in GPU Gems 2, M. Pharr, Ed. Boston: Addison-Wesley

Wagenbreth, G., Yao, K-T., Davis, D., Lucas, R., and Gottschalk, T., (2005), Enabling 1,000,000-Entity Simulations on Distributed Linux Clusters, WSC05-The Winter Simulation Conference, Orlando, Florida,

Whaley, R.C. & Dongarra, J.J., (1998), Automatically Tuned Linear Algebra Software, IEEE/ACM Conference on Supercomputing, SC98., pp.:38 - 38

Yao, K-T., Ward, C. & Wagenbreth, G., (2006), Agile Data Logging and Analysis, 2003 I/ITSEC Conference, Orlando, FL