

Enhancing Image Processing Performance for PCID in a Heterogeneous Network of Multi-core Processors¹

Richard Linderman

*Air Force Research Laboratory, Information Directorate,
AFRL/RI, 525 Brooks Road, Rome, NY 13441
Richard.Linderman@rl.af.mil*

Scott Spetka, Susan Emeny, Dennis Fitzgerald

*ITT Advanced Engineering and Sciences
AFRL/RITB 525 Brooks Road, Rome, NY 13441
Scott.Spetka.ctr@rl.af.mil, Susan.Emeny.ctr@rl.af.mil, Dennis.Fitzgerald.ctr@rl.af.mil*

Abstract— The Physically-Constrained Iterative Deconvolution (PCID) image deblurring code is being ported to heterogeneous networks of multi-core systems, including Intel Xeons and IBM Cell Broadband Engines. This paper reports results from experiments using the JAWS supercomputer at MHPCC (60 TFLOPS of dual-dual Xeon nodes linked with Infiniband) and the Cell Cluster at AFRL in Rome, NY. The Cell Cluster has 52 TFLOPS of Playstation 3 (PS3) nodes with IBM Cell Broadband Engine multi-cores and 15 dual-quad Xeon head nodes. The interconnect fabric includes Infiniband, 10 Gigabit Ethernet and 1 Gigabit Ethernet to each of the 336 PS3s. The results compare approaches to parallelizing FFT executions across the Xeons and the Cell's Synergistic Processing Elements (SPEs) for frame-level image processing. The experiments included Intel's Performance Primitives and Math Kernel Library, and FFTW3.2..

Optimization of FFTs in the PCID code led to a decrease in relative processing time for FFTs. Profiling PCID version 6.2, about one year ago, showed the 13 functions that accounted for the highest percentage of processing were all FFT processing functions. They accounted for over 88% of processing time in one run on Xeons. FFT optimizations led to improvement in the current PCID version 8.0. A recent profile showed that only two of the 19 functions with the highest processing time were FFT processing functions. Timing measurements showed that FFT processing for PCID version 8.0 has been reduced to less than 19% of overall processing time. We are working toward a goal of scaling to 200-400 cores per job (1-2 imagery frames/core). Running a pair of cores on each set of frames assigned to a worker reduces latency by implementing multithreading FFT processing. These results support the next higher level of parallelism in PCID, where groups of frames each producing one resolved image are sent to cliques of cores in a round robin fashion.

Current efforts toward further performance enhancement for PCID are shifting toward using the Playstation3s in conjunction with the Xeons to take advantage of outstanding price/performance as well as the Flops/Watt cost advantage. We are fine-tuning the PCID parallelization strategy to balance processing over Xeons and Cell BEs to find an optimal partitioning of PCID over the heterogeneous processors. A high performance information management system that exploits native Infiniband multicast is used to improve latency among the head nodes. Using a publication/subscription oriented information management system to implement a unified communications platform makes runs on large HPCs with thousands of intercommunicating cores more flexible and more fault tolerant. It features a loose coupling of publishers to subscribers through intervening brokers. We are also working on enhancing performance for both Xeons and Cell BEs, by moving selected operations to single precision. Techniques for adapting the code to single precision and performance results are reported.

TABLE OF CONTENTS

1. INTRODUCTION.....	2
2. MULTICORE OPTIMIZATION APPROACH.....	4
3. INFORMATION MANAGEMENT FOR PARALLELIZATION AND STREAMING.....	7
4. RESULTS.....	8

1

¹ U.S. Government work not protected by U.S. copyright.

5. FUTURE WORK 10
6. CONCLUSIONS 10

1. INTRODUCTION

The Physically-Constrained Iterative Deconvolution (PCID) algorithm is a multi-frame blind deconvolution algorithm developed for removing the image blur from atmospheric effects when observing space objects from the telescopes on the ground. The quality of the output produced by the algorithm has been shown to achieve or closely approach the associated Cramer-Rao lower bounds. A detailed discussion of the algorithm can be found in [1].

PCID processes sets (tens to hundreds) of blurred image frames into highly resolved reconstructed images through an iterative multi-frame blind deconvolution (MFBD). Fig. 1 shows some typical measurement frames from ground based observations of the space shuttle that were processed into the restorations in Fig. 2. Sensors typically capture such images at rates of 30 to 250 Hz as objects pass overhead for typically more than 5 minutes.

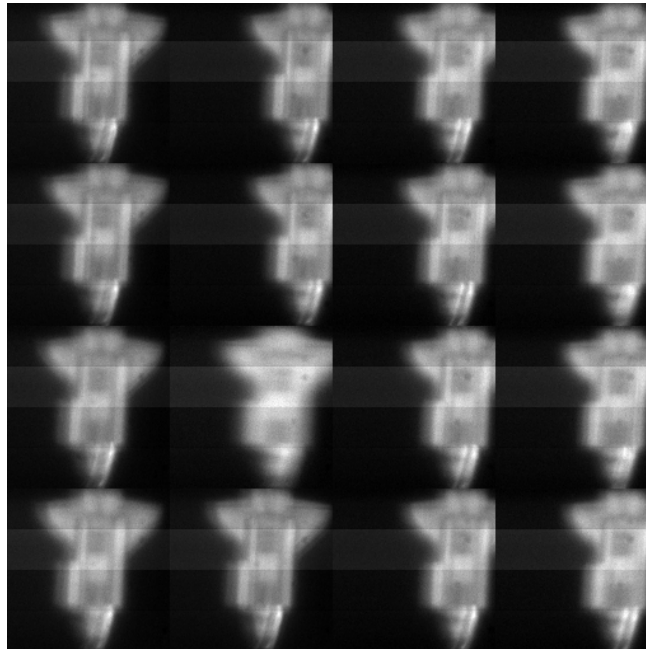


Fig. 1. Measurement frames of the Space Shuttle.

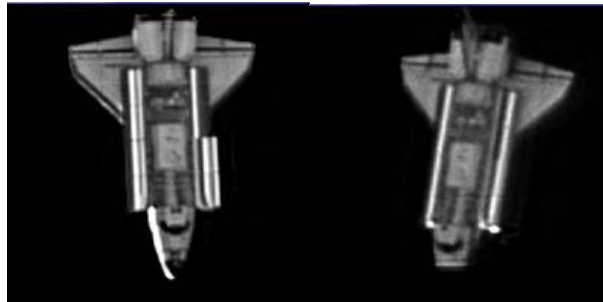


Fig. 2. Two image restorations of the Space Shuttle produced by the PCID algorithm.

Before recent optimizations, PCID runtime was dominated by many two dimensional Fast Fourier Transform (FFT)

calculations involved in image convolution. These were sped up by tuning the usage of optimized library functions and then parallelized effectively across Intel Xeon multi-core architectures (Woodcrest and Harpertown versions) as reported in [5].

A year ago, the baseline code running FFTW3.1 was achieving approximately 700 MFLOPS on the Xeon cores. FFTW3.2 was first examined, with difficulty attaining the benchmark performance numbers when calls were made from within the PCID code. Attention then shifted to the Intel IPP and MKL libraries from Intel. The results, in sustained GFLOPS, of optimizing two dimensional FFTs on the XEON processors with the IPP library are shown in Fig. 3. The nodes have dual Harpertown Xeons, each with four cores and 12 MB of Level 2 cache. The main sizes of interest are from 128x128 to 512x512. The results show that these sizes achieve excellent efficiency on the Xeons due to their large level 2 caches. Up to 1Kx1K FFT sizes, configuring the processors to run in pairs of cores (ones that share a 6 MB L2 cache is critical) and running four pairs simultaneously on separate FFTs gave the best performance. However, the impact of reaching beyond L2 cache is evident, and for 2Kx2K transforms it becomes more beneficial to allow all eight cores to work together on a single FFT.

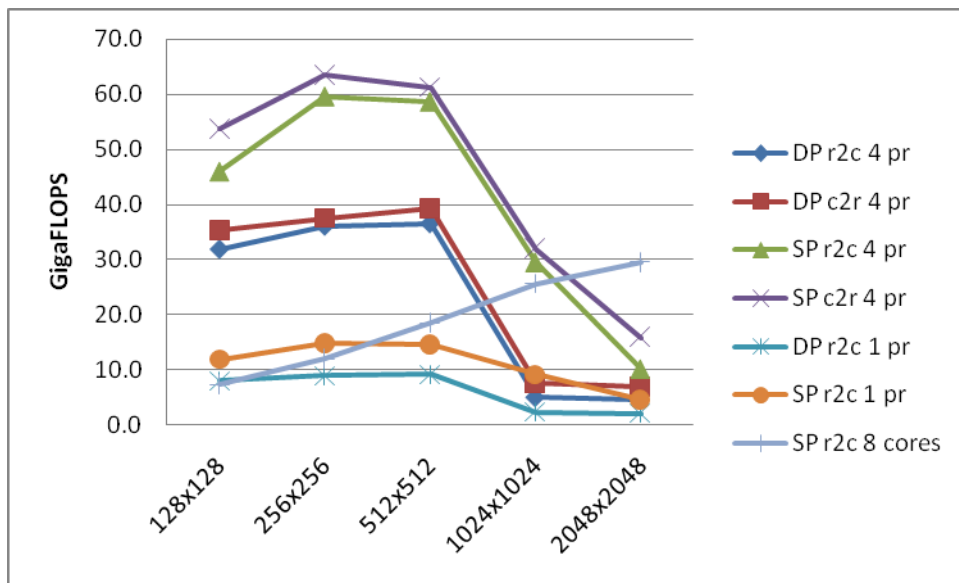


Fig. 3. Xeon FFT Results With Intel Math Kernel Library (MKL)

As a result of the performance optimizations, for small sized images such as the 128x128 images of the shuttle above, FFTs only consume 10-20% of runtime. However, for larger images they remain a major portion of the runtime.

Three levels of parallelism allowed PCID to make use of the many processors on cluster HPCs. Each set of image frames destined to produce a reconstructed image can be directed to a different clique of processors. The frames can be spread amongst processors within the clique. Finally, multiple processing cores within a node can work together on the two dimensional Fast Fourier Transform (FFT) calculations.

The objective of aggressive parallelization of the PCID code is to drive down the latency of forming images from minutes to a few seconds and to support a throughput adequate to sustain a real-time flow of images off of the sensor. The approach to this is discussed in section 2. Section 3 then discusses the transformation of the PCID code from MPI to a more flexible publish-subscribe-query basis of managed information objects. Section 4 gives results of the work to date, and section 5 outlines future work.

2. OPTIMIZATION AND PARALLELIZATION APPROACH

The approach is to first reduce the latency of FFT operations by optimizing usage of vendor libraries and applying more than a single core to these computational intensive tasks. The second step is to optimize other processing functions as well as improve the time needed for communications amongst the worker nodes of the clique and the clique master. The third step is to spawn multiple cliques to handle batches of incoming image frames in a round robin fashion and persist from assignment to assignment to minimize startup time.

With these three possible levels of parallelization, the PCID algorithm is well suited to taking advantage of very large high performance computers (HPCs). If low latency were not a driver, multicore optimization of functions like FFT would be undesirable. Instead, each core could simply be an independent member of a processing clique, assigned unique frames of input imagery. However, when low latency image production is important, going to the extra effort of assigning a small number of images to each multi-core and threading the 2D FFT computation across cores pays dividends. As mentioned in the background, FFT optimization results indicated that two cores sharing a level 2 cache and working together on FFTs was the best solution for Woodcrest and Harpertown Xeons accept for larger FFTs sizes of 2048 or more. The next levels of parallelism involve how many pairs work within a clique to produce an image, and how many cliques operate in parallel to produce a stream of resolved images. The proper sizing depends on the architecture of the HPCs employed.

Multi-core High Performance Computers Studied

There are a variety of multi-core architectures emerging in the market. In this work, the focus is primarily on Intel Xeon® processors and the IBM Cell Broadband Engine® as found in the Playstation3® gaming consoles as well as the largest supercomputer, Roadrunner, at Los Alamos. The optimization and porting work has been conducted on two HPCs, the 1280 node JAWS cluster at the Maui DoD Supercomputing Resource Center and the 336 node Cell Cluster at the AFRL HPC center in Rome, NY.

JAWS (Fig. 4) features 1280 Dell nodes each with two Woodcrest 3 GHz dual core Xeons, 8 GBytes of memory, and an Infiniband networking fabric.



Fig. 4. Maui HPC Center JAWS 1280 node Dual-Dual Xeon Cluster

The Cell Cluster is a hybrid cluster of 14 subclusters. Each subcluster has a headnode with two quad-core 3 GHz Harpertown Xeons. These feature 12 MB L2 cache, 32 GB of memory, and both infiniband and 10 gigabit ethernet fabrics. Under each headnode are 24 PS3s. Each PS3 features 153 billion single precision operations per second (GFLOPS) from 6 cores behind a PowerPC frontend core. The Cell Broadband Engine actually has 8 backside cores, but only 6 are available to the user in the PS3. Each PS3 has just 256MB of memory and a gigabit ethernet interface. The intent of this heterogeneous architecture is to allow the memory and I/O strengths of the subcluster headnodes to compensate for the PS3 capabilities below them. Fig. 5 shows the Cell Cluster network. Red lines are 10 gigabit Ethernet and black lines from the headnodes to the central switches are 4X Infiniband. Fig. 6 shows the

racking of the PS3s in the lab. The PS3s were left in their original enclosures to preserve some system engineering advantages and save costs of repackaging. The bread racks provide both portability and airflow up through the 48 PS3 on each rack. The Cell Cluster delivers outstanding price-performance approaching 200 teraflops per million dollars. More information on the Cell Cluster and earlier applications results can be found in [2].

Both machines run the linux operating system. In the case of the PS3s, Linux (Fedora or Yellow Dog) runs on top of the Sony Hypervisor. Both the PS3s and the headnodes also provide the Message Passing Interface (MPI) library and optimized math libraries for accelerating functions like the FFTs. JAWS has a total of 5120 Xeon cores for 62 Teraflops peak performance. The Cell Cluster 52 Teraflops from its 2016 cores within the Cell BE chips.

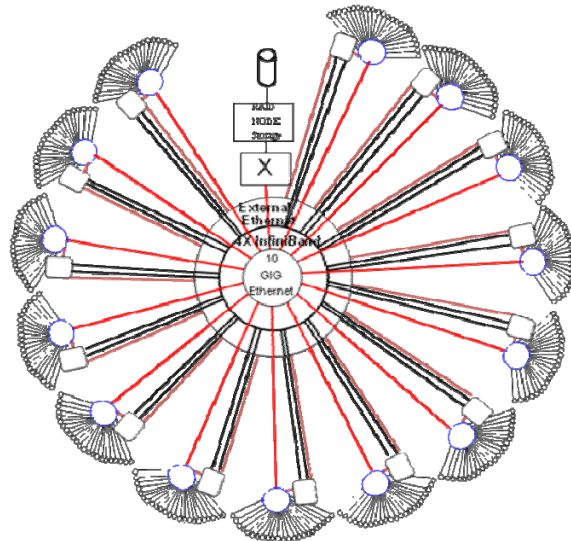


Fig. 5. Cell Cluster Networking



Fig. 6. Cell Cluster in HPC Laboratory

Clique Approach to Large-Scale HPC Parallelization

PCID can operate on inputs from a variety of sensors. Typically, raw images vary from 128x128 pixel arrays up to 1kx1k. Frame rates also vary considerably with some of the smaller sensors acquiring at 250 Hz, and some of the larger sensors around 30 Hz. Another key variable is the number of raw image frames that go into formulating a resolved image which can vary widely from the neighborhood of 20 to 500. The number of frames/image

reconstruction is limited by what is available, but also by forming set with similar view points where factors such as magnification remain fairly constant. A typical overpass of an object might include six minutes worth of frames. So for example, 128x128 pixel images could be acquired at a 200 Hz rate and an image generated from each set of 200 images by dispatching the incoming frames in a round-robin fashion to cliques of processors or cores. For a given number of cores, there is a direct tradeoff between cores/cliقة and numbers of cliques. The tradeoff is that devoting more cores/cliقة reduces the latency of producing the image, but at an expense of processing efficiency since the intra-cliقة communication detracts from perfect scalability with clique size. If latency is not an issue, then running PCID with very few cores per clique is beneficial, as long as node memory suffices. But latency is of growing importance, motivating larger cliques and efforts to improve the scalability of the code across nodes. Given that cores are run in pairs to improve performance based on multithreaded libraries, the clique size can vary from 1 pair (serial case) to a maximally parallel case of 1 pair per raw input frame. There is some algorithmic inefficiency in going to this extreme parallelism since some computations, such as the FFT of the object estimate, can be performed once and used across all frames assigned to a node. However, internode I/O usually limits desired clique size well before this maximal case becomes an issue. The internodal I/O is also a function of the minimizer employed. For example, the conjugate gradient (CG) minimizer requires much less I/O than the L-BFGS-B minimize. The parallelization studies focused on clique sizes from 1 to 201 pairs when considering a 400 frame case using the CG minimizer. The cliques contain one master pair and N worker pairs (hence 201 applied against 400 frames gives each worker two frames).

Incoming batches of frames are assigned to available cliques on an as available basis. In a real-time system, frames would have to be dropped or archived for later processing if no clique was available at the time. After capturing a particular overpass for several minutes, it is reasonable to assume that the telescope acquires another object with different processing parameters which will likely imply a different organization of its cliques. Hence dynamic and flexible information management system is needed to efficiently apply HPC resources against the inputs. This is discussed in section 3.

Further FFT and Other Optimizations and Single Precision Considerations

FFT optimization results for the Harpertown Xeon processors were shown in Fig. 3 and are further discussed in [5]. For the PS3, the initial FFT support came from FFT3.2. However, recent results under the SPIRAL system[3] seem likely to further push 1D FFT performance toward achieving 50% of peak performance. The SPIRAL FFT support is presently being extended to include two dimensional real-to-complex FFTs and complex-to-real inverse FFTs. These new functions will then be integrated into the PCID code.

The original PCID code performs all of its calculations in IEEE double precision floating point precision (64 bit). However, single precision, or a mixture of single and double precision may suffice. This change to single precision has roughly a 2X performance improvement on the Xeon cores, but in theory as much as a 14X improvement on the PS3 cores. The Cell BE processors in the PS3 are not pipelined for double precision. As an aside, the DP arithmetic is pipelined in the variant of the Cell BE used in the Roadrunner supercomputer at Los Alamos. Beyond runtime performance, an additional benefit of moving to single precision where possible is that the memory footprint is cut in half. This is helpful on the Xeon nodes, but even more helpful on the PS3s which have 256 MB of DRAM.

Since the present PCID code uses double precision exclusively, the first step in the approach was to make a wholesale conversion to single precision in order to bound the speed improvements, image quality degradations, and convergence speed impacts. Subsequent mixed precision approaches can then move us to a “middle of the road” position that balances these tradeoffs.

The initial conversion was done on a Phantom headnode of the Cell Cluster and on Jaws. One of the Phantom headnodes serves as the “Master” process for the Cell Cluster port of PCID, with the PS3s acting as the “workers”. The single test case was 400 frames of shuttle imagery with 256x256 Gemini data run through PCID for 100 iterations. A broader set of reconstructions will be run to augment these initial results.

Run serially on Jaws, the DP code required 291 seconds while the SP code required only 148 seconds—the anticipated near doubling in speed was observed. The convergence of the two runs was nearly identical across the

100 iterations of the code. The resulting images are very similar as shown in Fig. 7. While more image comparisons need to be performed, this initial work indicates good prospects for single precision delivering good image quality in at least some cases.

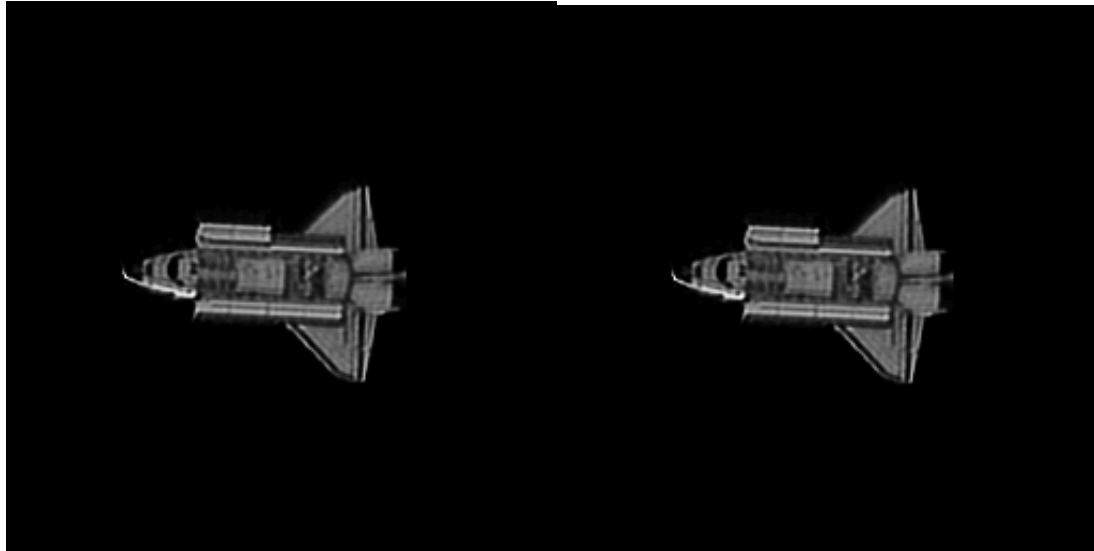


Fig. 7. Double precision (left) versus single precision (right)

3. INFORMATION MANAGEMENT FOR CLIQUE CONTROL

The need for information management services more robust and flexible than point-to-point intercommunications arose far from the HPC realm. The desire for business to business collaborations and connections of complex systems across the internet was one driver for a loosely connected publish and subscribe paradigm. In this realm, publishers put out their information objects without knowing how many subscribers exist. The intermediary services of a broker obviate the need for the publisher to be concerned with such dissemination details. This is especially helpful in a dynamic (or fault tolerant) environment where subscribers are expected to come and go. When the broker arranges to archive the information objects, a query service can be added so that subscribers can access objects published in the past.

These concepts have undergirded a “Joint Battlespace Infosphere” (JBI) concept pursued by the Air Force Research Laboratory [4]. The basic approach is to create information objects by marking up “payloads”, which may be binary objects like data arrays or images, with XML metadata. The subscriptions and queries are then reflected as XPATH predicates evaluated on the metadata of the information objects from either publishers, or the archives respectively.

Publish, subscribe and query services offer to HPC users a new paradigm for programming that is loosely coupled for both flexibility and fault tolerance, persistent across runs, and managed to provide consistent quality of service and security to users. This programming paradigm can either replace current methods of interprocessor communications, notably MPI, or work alongside them. While the flexibility and fault tolerance are increasingly desired within HPCs as core counts grow, performance remains the key issue. To be acceptable to the user community the services have to be performed at a reasonable overhead relative to the MPI services. The present version of the information management environment can tens of millions of information objects per second within a thousand core HPC with simple predicates as our typically found in MPI programs. The latency of message delivery is on the order of 60 microseconds for unicasts.

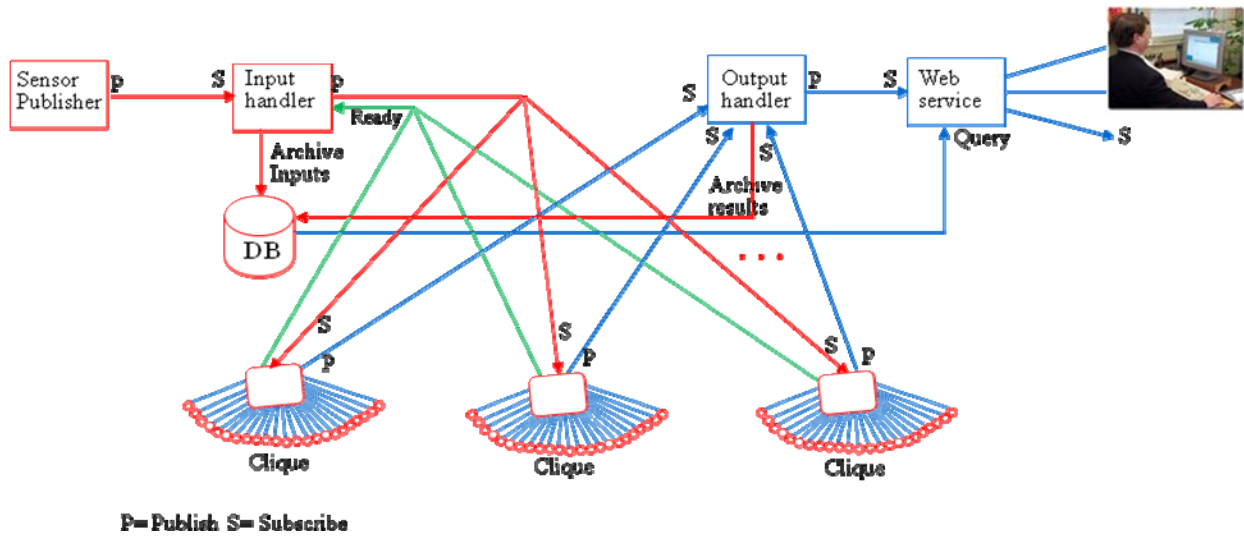


Fig. 8. PCID Multicore Flow of Information Objects

Fig. 8 depicts the publish-subscribe-query formulation of the PCID application in a real-time mode. The imaging sensor delivers frames to an input handler which makes assignments on a round robin basis to the cliques that have indicated they are ready for an assignment. If a clique is delayed, it is simple to assign another ready clique to the task at hand. This is particularly helpful, since PCID is an iterative algorithm without a highly deterministic runtime. The assignment is made by setting clique ID fields in the metadata of the published image. The frame sequence number then steers particular frames to particular processors. The input handler can remain unaware of the details of the mapping of frames to processing nodes.

The master of each clique controls the iterative deconvolution process by subscribing to partial results from the clique members and then multicasting instructions for the next iteration to its clique. Upon converging to the resolved image, the master publishes the result to the output handler and marks it for archiving—to support future queries.

The output handler is coupled to a web service the can disseminate the live results to multiple users over the internet.

4. RESULTS

Over the past year, the runtime of parallel version of PCID has been improved through a series of optimizations discussed above and some algorithmic refinements eliminating unnecessary work. Fig. 9 shows the comparative runtime of PCID version 7.3 from September, 2008 compared to recent runs of PCID version 8.0 on Jaws and the Phantom headnodes of the Cell Cluster. These results are for a test case with 400 frames of space shuttle imagery processed through 100 iterations on varying sizes of cliques. As the FFT performance was dramatically increased, its percentage of overall runtime decreased from 88% to less than 20%. Processing time was further reduced by calling other MKL library functions. The intra-clique message passing has now become a more significant percentage of runtime due to much less time spent on computations. In addition, the serial startup time at the beginning of the job, approximately 8 seconds in this case, becomes a limiting factor needing parallelization.

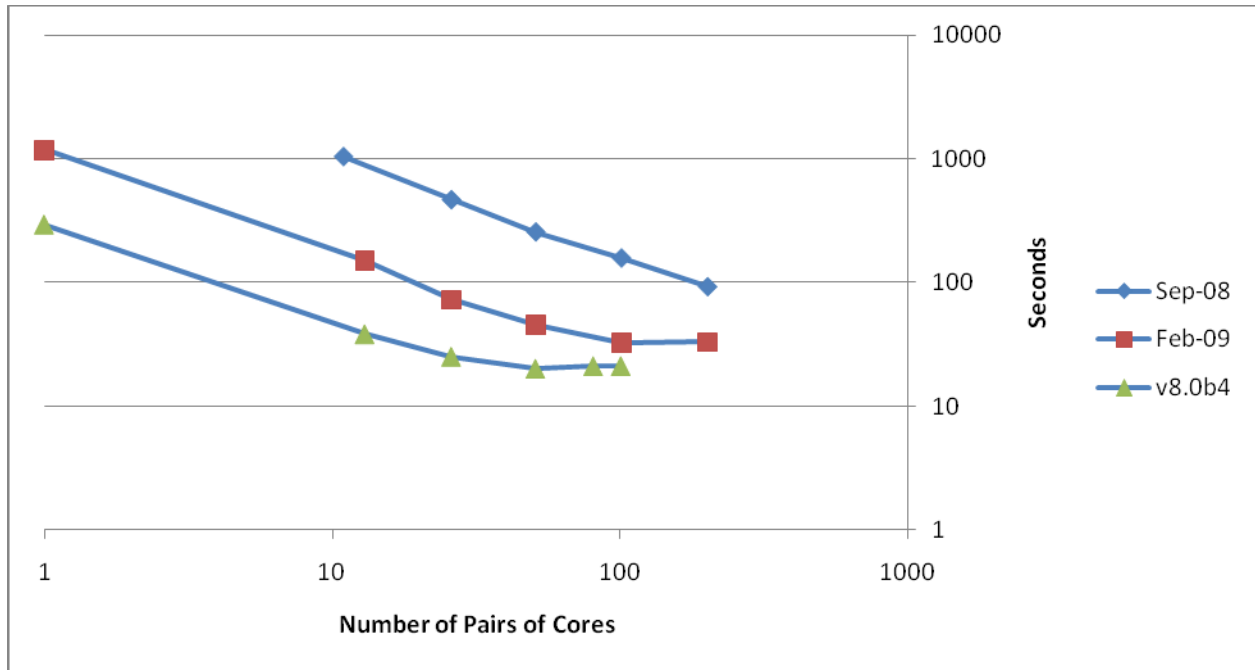


Fig. 9. 400 Frame, 100 iteration, 128x128 image benchmark scalability results

The input handler and output handler were built with the JBI publish-subscribe information management infrastructure and drove complete overpasses into various configurations of cliques to explore the latency-throughput tradeoffs. Two test cases were used as described in the following table.

Sensor	Frame rate (Hz)	Frame size	fr/image	Duration (seconds)	Iterations	Embed in 2x larger?	Reconstructed Images/pass
AEOS	35	512x512	175	5	100	No	60
Gemini	250	128x128	200	0.8	60	Yes	400

Initial results in double precision show the 60 image AEOS overpass completing in 1286 seconds when using 10 cliques of 51 pairs (102 cores) each. The average latency to produce an image was 199 seconds. For the Gemini test case the 128x128 raw images are embedded within 256x256 images to eliminate aliasing effects in the FFTs at the expense of computational time. When 40 cliques of 12 pairs each (960 total cores) work the problem, wall clock time to produce 400 images is 485 seconds with an average latency of 43 seconds. When 19 cliques of 26 pairs (988 cores) attack the problem, latency is driven down to 31 seconds at the expense of wall clock time increasing to 672 seconds.

The code was ported from double precision to single precision on Jaws with the doubling of performance mentioned above. The code was then ported to the PS3. In initial PS3 testing, the FFTW library calls are only delivering 700 MFLOPS, but are functioning correctly.

5. FUTURE WORK

SPIRAL FFT software from Carnegie-Mellon needs to be examined to potentially accelerate XEON performance over the MKL library performance. However, SPIRAL will likely provide significant needed improvements to the PS3 performance of the code.

Further experimentation on numerous test cases is required to ascertain the conditions under which single precision floating point suffices for PCID, and where this is found inadequate, mixed precision approaches need to be pursued.

Improvements to the collective communications within PCID, which present limit scalability on JAWS need to be explored. These include examination of MPI broadcast and reduce functions, as well as replacement of these calls with publish/subscribe alternatives under the JBI.

Finally, the design of Fig. 8 needs to be expanded to include the real-time sensor interface on the left, and the HPC-backed service made available on the internet to users on the right.

6. CONCLUSIONS

The Physically Constrained Iterative Deconvolution (PCID) algorithm allows for three levels of parallelization and can scale well to make effective use of large cluster HPCs in the 50 teraflops class. The advent of multicore architectures has motivated a shift from a pure message passing implementation with MPI to a hybrid approach. MPI message passing at the upper level can be effectively replaced with publish-subscribe services that allow for flexible construction and fault tolerance. At the lowest level, the 2D FFTs can be threaded across multiple cores with optimized libraries to reduce latency exploiting the shared memory and caches.

Wholesale conversion of PCID from double precision to single precision works well in at least some test cases and delivers a performance doubling on Xeons. This opens the door to even greater acceleration of the code on emerging multicore architectures.

REFERENCES

1. Matson, C.L. et al, A Fast and Optimal Multi-Frame Blind Deconvolution Algorithm for High Resolution Ground-Based Imaging of Space Objects, *Applied Optics*, April 2008.
2. Linderman, R. "Early Experience with Algorithm Optimizations on Clusters of Playstation 3's" Proceedings of the Users Group Conference (DOD_UGC'08), July 2008.
3. Voronenko, Y., et al, "Generating High-Performance General Size Linear Transform Libraries Using Spiral," High Performance Embedded Computing 2008, pp. 133-134, September 2008.
4. Combs, V., Linderman, M., "A Jini-Based Publish and Subscribe Capability", Proceedings of SPIE, Volume 4863, Java/Jini Technologies and High-Performance Pervasive Computing, June 2002, pp. 59-69.
5. Linderman, R., et al, "Parallelizing a Multi-Frame Blind Deconvolution Algorithm on Clusters of Multicore Processors". IEEE Aerospace 2009, March 2009, pp. 1-7.

BIOGRAPHY



Dr. Richard Linderman is the Air Force Senior Scientist for Advanced Computing Architectures. He works at the Information Directorate of the Air Force Research Laboratory in Rome, NY. His area includes high performance computers, embedded computer architectures, fault tolerant architectures, distributed and

next generation architectures.

Dr. Linderman is an IEEE Fellow and an AFRL Fellow. Dr. Linderman received his PhD from Cornell University in 1984 and taught at AFIT until 1988, when he joined the Laboratory.



Dr. Scott Spetka is a professor at the State University of New York Institute of Technology and a principal software development engineer at ITT Corp. He has worked onsite at the Air Force Research Laboratory Information Directorate since 1993. His work has been in the area of high performance computing for the past seven years. In the 1980s, he worked on developing LOCUS, one of the earliest distributed operating systems. He has a B.S. in mathematics from Denison University and a Ph.D. in computer science from UCLA.



Dennis Fitzgerald is a program manager for ITT Advance Engineering and Sciences where he manages a group of engineers specializing in High Performance Computing. He has developed various types of software including compilers, simulators and other development tools. He has experience optimizing and parallelizing software to maximize performance. He has a BS in Computer Science from Syracuse University and an MS in Computer and Information Science from the State University of New York.



Susan Emeny is a software engineer with ITT Advanced Engineering and Sciences. Susan has a wide range of software development experience including operating system development, signal processing, controls, and parallel processing. Most recently, Susan has been involved with High Performance Computing through her work with the Air Force Research Laboratory in Rome, NY. Susan has worked on software performance optimization on various platforms such as power PC, Linux clusters and IBM Cell processors. Susan has a BS in Mathematics from the State University of New York at Brockport.