89 Fifth Avenue, 7th Floor

New York, NY 10003

www.TheEdison.com

@EdisonGroupInc

212.367.7400

White Paper

Trading Smarter: How GPUs Improve Data Analytics and High Performance Computing in Financial Services

Printed in the United States of America

First Publication: September 2015

Produced by: Matthew Elkourie, Analyst; Manny Frishberg, Editor; Barry Cohen, Editor-in-Chief

Table of Contents

Executive Summary

Big data presents new challenges to institutional and enterprise customers alike, demanding more complex computational analysis and interpretation. Trading firms face unique challenges from several factors.

On the sell side, increased regulation has forced even more demanding risk reporting like comprehensive capital analysis and review (CCAR), and a variety of stress tests. These types of reports make huge additional demands upon existing compute grids, driving users to consider growing their compute grid capabilities significantly.

High-frequency traders, whether on the buy or sell side, are increasingly focused on trading smarter, rather than merely trading faster, by considering additional factors and larger amounts of data in strategy models. The amount of data being analyzed under this work-smart approach is growing exponentially. Traditionally, operations that relied on high performance computing (HPC) have relied on Moore's Law, the doubling of transistor density every 18 to 24 months, and subsequent improvements in either CPU core speed or core count.

While increased core speed and processor evolution itself continues, a shift is underway in how Moore's Law is applied to performance in compute grids. Massively parallel systems, driven by increasing core density per chip and tightly coupling co-processor options like the use of GPUs, are overtaking the use of raw processor speed and core count as a performance metric. Dramatically increased core counts available via GPU-enabled systems also bring increased complexity and new software challenges.

GPUs add considerable processing power to already capable nodes, but add challenges to both hardware, with unique infrastructure and systems design requirements, and the ability of software to run efficiently and leverage their benefits. Users of GPU-enabled architectures have to balance achieving maximum raw performance against programming ease and productivity.

In this competitive landscape, financial services (FS) decision makers are weighing an increasing number of variables. This short white paper aims to provide much needed guidance on the challenges and use of GPUs in FS. Edison will take a brief look at the hardware and software challenges of GPUs and evaluate NVIDIA's solution for GPU-enhanced HPC platforms. Then, the report will take a close look at how Cray has

addressed the same concerns and challenges, by evaluating the features and advantages of the Cray® CS-Storm™ system, a fully integrated Cray and NVIDIA GPU offering.

Scaling Up in Financial Services

How can the speed of decision-making be increased? First, by speeding up the search for and modeling of alpha-generation ideas, and second by speeding the execution of those ideas in market-facing systems. Speeding up the modeling demands systems that are throughput-capable and can process both ever increasing amounts of structured market data and unstructured "secret sauce" data such as news feeds, weather, and wellhead data. The second demands low-latency systems that can generate trading signals from a minimum of model input data. Accelerator technology can help relieve CPU bottlenecks to significantly improve performance.

Can additional details be factored in by analysis or backtesting of market data? What technologies will ensure leadership in the long run? FS analysts ask these questions constantly.

The quest for faster, better, and more data is evolving as computers and the software that run on them are continually refined to meet them.

Historically, as a category, HPC has defined the performance index of a "system" and its associated speed, loosely, by measuring floating-point operations per second (or flops). Much like hard drives and the dramatic growth in storage capacity, system CPUs have become increasingly sophisticated. Whereas not so long ago, measuring overall system performance in teraflops was adequate, today it is not uncommon for a single node contained within a system to exceed 1 teraflop.

Execution speed, efficiency, and accuracy are critical factors when considering the workloads typically faced by FS analysts. The pressure to perform quickly can be felt across a wide variety of use cases, including derivative pricing, value at risk (VAR), CCAR stress tests, and credit valuation adjustment. Market data that bears on all these use cases is growing, not shrinking.

A good example is high-frequency trading, where the ability to rapidly sift through transient market data in real time is crucial. Rather than having more time to deliver increasingly complex and critical results, less time is available, and more data is required to make better decisions.

The industry is trending toward trading smarter, versus simply trading faster. While speed is important to some FS applications, the quality of the analysis is equally important. Traditional approaches would include increasing overall node count and

availability, but enhancing existing systems or adding new infrastructure with coprocessors or GPUs should also be considered. HPC systems like the Cray CS-Storm supercomputers with NVIDIA GPUs are well suited to meet these increased hardware demands.

Achieving peak performance also requires the right software to compile, distribute, and execute FS specific applications. The variety of parallel programming options include (but are not limited to) CUDA® (Compute Unified Device Architecture), OpenACC (Open Accelerators), OpenCL (Open Compute Language), and TBB (Thread Building Blocks). FS analysts must find the right balance between pure execution speed and the ease with which programmers can develop, deploy, migrate, and manage FS applications and code base(s).

The State of GPUs in HPC

While computers have arguably managed to keep pace with Moore's Law in terms of speed, the ways in which we define current and future performance are shifting from raw CPU speed toward system parallelization.

In the past, one might simply add hardware to solve increasingly complex and expanding problems. As the industry shifts toward a compute-node parallelization model, more sophisticated programming models are needed to keep up with the greater complexity of the hardware.

When addressing traditional HPC and increasing execution speeds in the FS sector, the main considerations in determining how quickly jobs can be executed and completed would be core count and the speed of the system's processors. While this thinking is important, FS is already used to perform derivative pricing and risk calculations being highly parallelized for many assets in a portfolio, across thousands of Monte Carlo paths. While adding additional nodes is a reasonable way to increase job throughput, a better alternative is increasing compute density at the node, using graphical processing units (GPUs) to take advantage of the massive parallelization this technology offers.

GPU usage itself is not a new concept in HPC. In fact, the term general purpose computing on graphics processing units (GPGPU) dates to 2001, when usage of GPUs operating alongside CPUs became popular as efforts to perform linear algebra implementations on GPUs ramped up.

As GPU usage in HPC has grown, so has the challenge GPUs pose in providing node-integrated platform and useable software environment to develop or port code. NVIDIA, a leader in the GPU manufacturing and development markets, develops both the hardware platforms on which GPUs are hosted (currently the NVIDIA® Tesla® family), as well as CUDA, one of the software platforms required to develop, integrate, and execute code on these systems.

As GPU systems are coprocessors to a traditional Intel CPU, data has to be moved from the CPU to the GPU to execute and the results moved back. The overall performance is therefore limited by the amount of GPU memory and the bandwidth of the data bus between CPU and GPU.

With continued development by NVIDIA to increase GPU memory size, GPU-to-CPU memory bandwidth, and GPU-CPU memory integration, more latency-sensitive

applications have become viable on GPUs. Changes in the frequency of market data evaluation, as well as the increased ability to evaluate larger amounts of data paths, has enabled FS to shift from just trading faster to trading smarter. GPU-enabled systems, with their ability to deliver massive core availability per node, make an excellent partner for FS, helping it scale up to meet increased demands for data analysis.

GPU Facts and Challenges

Thus far, Edison has explored areas in FS where GPU usage can greatly boost productivity. GPUs clearly offer performance advantages with applications and toolkits built to leverage the massive performance capability of on-node parallelism. NVIDIA's current generation of Tesla GPUs, for example, offers up to 4,922 cores and 24 gigabytes of memory per GPU, translating into astounding performance that measures up to 1.87 teraflops of double precision performance per GPU.

Applications and code have also continued to evolve to take advantage of the extended capabilities GPUs and similar coprocessor solutions for the FS market. Once deployed, GPU-enabled systems can enter production rapidly, performing the same tasks previously run by FS institutes at a higher speed, using parallel programming software like OpenACC (and eventually OpenMP) or others.

So where's the rub? Surely if GPUs were the holy grail to HPC, and FS specifically, why is the entire industry not transitioning code and operational infrastructure to GPU-enabled systems? The answers can be found in the unique challenges to successfully deploying the physical GPU stack, as well as the programming challenges in both porting, and then efficiently running code specific to the job at hand.

The Software Challenge

Due to the complex nature of FS applications, figuring out whether GPUs make sense from a hardware standpoint is not enough. Decision makers must also evaluate both the software and the end user or programmer's ability to port, manage, and execute applications on the GPU-enabled stack.

Tremendous changes have been made in CPUs over the last decade. CPUs have scaled horizontally by increasing the number of cores in a node. They have also added and widened vector units. In order to take advantage of these developments and not leave CPU capability on the table, applications need to take advantage of thread-level and vector parallelism. Critical thinking by the software designer is necessary to take advantage of GPUs. In addition to parallelizing applications, critical thinking for GPUs needs to be applied to the data transfer to the GPU and efficient use of the GPU's more limited memory.

Some common software-related questions include:

- What current applications are (or can be) parallelized to take advantage of the GPU architecture?
- Can today's compilers and supporting code onboard and execute tasks via non-host devices (GPUs for example)?
- What are the staff's capabilities to enhance or modernize existing code and take advantage of significantly higher-density node-parallelism capabilities?
- Can programs also take advantage of single-node parallel hardware?
- What programming languages are staff familiar with?
- Can the high performing kernel codes (e.g. quant libraries) be isolated and reused, concentrating performance skills in a smaller group?
- Is the prime objective ease of use or squeezing the last bit of performance out of the HPC infrastructure?
- What other libraries are interacting in deploying and running applications (R, Python, MATLAB, etc.)?

For FS users that have already made significant investments in their HPC infrastructure, the approach taken might come down to a focus on peak performance, or a desire for software programming models to be more flexible and open to access by required libraries already in use.

Utilizing or migrating to a hybrid parallel programming approach over host-based programming can allow FS users to enjoy greater flexibility in choosing the hardware to be deployed. While it's true that host-based programming models can be tuned for maximum performance of a given platform, the hybrid parallel programming model supported in OpenMP and OpenACC (for example) opens the door to additional options in node compute capabilities with access to several coprocessor choices. A great example of why this choice can matter is provided by a look at OpenMP.

OpenMP 4.0 is relatively indifferent to the location of the coprocessor and is flexible with regard to peripheral software and associated hardware. With support expected for Intel's forthcoming CPU changes, and support for GPUs already present (among other coprocessor options), OpenMP's ability to hedge future hardware investments by leveraging "neutral" software can be alluring to some users. The tradeoff is giving up a slight performance advantage over dedicated software written for a specific hardware

structure (CUDA). For example, even where OpenACC is in use today it is common to first start by parallelizing applications, if they were not already, with OpenMP.

Similarly, users who have already adopted NVIDIA GPUs, and have on-boarded talent to code in languages like CUDA, may find themselves unable or unwilling to give up the performance advantages CUDA can deliver.

For some FS institutions, merely obtaining some help in bridging the technology gap between CPU-only based computing and GPU-enabled hosts can seem a daunting task. While OpenACC is mentioned previously as part of an OpenMP strategy, some of the attraction towards OpenACC is found in the framework's reduced coding complexity. Coupled with full compiler compatibility and support from mainstream companies like Cray and Portland Group, the OpenACC framework is a mature technology platform from which institutions can either test alternate hardware solutions or create a stop-gap to either CUDA- or OpenMP-based production applications if they so choose.

Although the choices in how GPU or coprocessor hardware is addressed, and jobs then dispatched, can vary programmatically, with available node parallelization capacity, application performance on GPU-enabled nodes can dramatically increase how typical FS HPC applications scale. Take Monte Carlo VAR calculations, for example – a workload that forms the bulk of FS daily activity. Monte Carlo applications are highly parallel, with the ability to scale as users add more cores, with less of an emphasis on per-node memory capacity and bandwidth.

These requirements are a good example of GPU-enabled HPC systems presenting the right solution when running Monte Carlo calculations. While the workload is large, it can be distributed into a collection of node-sized chunks, shrinking the time to complete the simulation significantly.

The Hardware Challenge

When looking at typical purchase decisions for HPC environments with the FS market vertical as a focus, a set of hardware-related questions emerges:

- How dense are the nodes to be purchased in terms of CPU/cores/memory?
- How much work can each node perform?
- How far can these nodes and related supporting architecture scale?
- How much do the nodes cost, and what is the system's total cost of ownership (TCO)?

- Where will these nodes be housed and maintained?
- What are the performance enhancements from a GPU-based solution, compared to traditional CPU-only based HPC solutions?

Evaluating all the possible answers to these questions could take up a research paper all by itself. Here are some available facts and figures that summarize some possible advantages GPU solutions bring to the table.

Density of CPU computational power at the node level currently is limited to a typical two- or four-socket design. While CPU core counts and chip manufacturing efficiencies will continue to rise for the foreseeable future, much of the micro-architecture of the CPU is devoted to fast scalar performance not extreme levels of parallelism. With thousands of simpler cores per chip and much less general-purpose circuitry, the performance advantages of GPUs for the highly parallel workloads typical in FS have already proven themselves in many production settings.

Consider this: A Cray CS-Storm system configured with integrated Cray and NVIDIA components can deliver up to 329 teraflops of double precision compute performance in a single rack. The CS-Storm system allows you to deploy up to 1 terabyte of CPU memory on node, with an additional 192 gigabytes of GPU memory available per node. The combination of processing power and memory size provides for some heavy lifting capabilities, allowing for serious scaling of parallel jobs.

Of course, the need to deliver fast performance would seem to imply that institutions would need to deploy very large clusters in order to stay competitive. Having a very large cluster becomes costly as a variety of factors like systems procurement cost, systems implementation and software programming requirements, provisioning of facilities (power, space, cooling), and other factors are weighed into total systems acquisition and commissioning.

Traditional cost calculations for HPC clusters involve the number of nodes needed to complete a given workload in a given time frame. In simplistic terms, if firms made an assumption that today's workload takes a certain amount of time over a given amount of nodes, to achieve tomorrow's results one would simply do the math to determine how many more nodes would be needed to achieve the objectives. Unfortunately, the answers are never that simple. A variety of factors, like space and power required, supporting infrastructure, and managing the size of the cluster all come into play.

GPUs can help resolve many of these issues. NVIDIA Tesla GPUs (in this case the K40/K80 family) help reduce significantly the amount of compute resources required at

the node level, providing up to 1.87 teraflops of double precision performance over 4,992 cores and 24 gigabytes of memory per GPU.

With the increased performance available to GPU-enabled systems and the ability of the Cray CS-Storm system to host up to eight of these GPUs per physical compute node, the space needed for the cluster is less of an issue. Further, with dramatically increased ability to perform work per node, users can perform larger workloads at less cost.

Challenge Summary

GPUs can be an excellent choice to reduce the cost of and speed up highly parallel grid application code, but the route followed by a fully integrated production GPU operation is complex, and needs to be considered carefully. The challenges are in application development, hardware infrastructure and building an expert support organization.

Cray sells more accelerated compute power than anyone else, and is well placed to advise in all aspects of the challenges, as well as to provide the best systems, designed to take full advantage of the GPU's power.

Cray's CS-Storm cluster systems can provide a clear path for accelerated performance in most user environments, providing the necessary stability that both coprocessor hardware and FS software choices require. A brief product summary is provided in Appendix A of this white paper.

GPU and Systems Design Considerations

While there are clear methods for deploying and utilizing GPUs, both physically and programmatically, what has not been discussed is the design process systems vendors must consider when offering GPU-based solutions.

As with any highly complex architecture, the advantages do not come without considering the design, implementation, and management of the HPC infrastructure. With GPUs, aside from typical HPC challenges, a few new factors are introduced that should be evaluated.

Anyone can simply plug a GPU into a system board (for example, via the PCI-E slot available to most systems) and arguably derive some measurable amount of performance increase. It is quite another challenge to offer GPUs as a coprocessor option in enterprise-grade, mission-critical production systems.

Systems integrators must consider several factors to achieve a working, reliable, and scalable product. The many variables include:

- Node form factor,
- Node reliability under stress,
- Systems integration of a working node farm inclusive of GPUs, compute resources, and related networking and storage resources,
- Hardware compatibility with anticipated user software (i.e. drivers, reporting metrics, and ability of the hardware to be fully exploited by the user software) or vendor-supplied tools to support proprietary vendor hardware,
- Systems architecture that provides sufficient resources, both internally (networking and storage access) and externally (power delivery and management, cooling), to allow 100 percent systems utilization at full load,
- And, vendor supplied software tools and libraries that help users who are already familiar with the tools and resources, or that otherwise reduce the learning curve in on-boarding to the GPU-enhanced cluster.

Engineering challenges like form factor, power and cooling, and systems integration are key factors to purchasing and implementation. Normally, vendor engineering decisions affect how the GPU-enabled system will be delivered, implemented, and deployed at the user's site.

In most GPU-enabled systems, the concept of GPU delivery is accomplished by locating the GPU component externally and then connecting this to the node or nodes that will access the GPUs. More recently, GPUs are being fully integrated within the compute node infrastructure.

An important design consideration for GPU systems is efficient cooling. GPUs run hot and are designed to throttle back their performance if they run too hot. In the rush to produce GPU systems as their use increases, many vendors have not focused enough on proper whole-rack cooling. As a result, with hard use their systems produce disappointing results. In order to properly evaluate a system's performance, it is not enough to look at peak teraflops. A system should be evaluated under full load to ensure it is not susceptible to temperature-driven internal throttling.

Another factor is that hot GPU's are more prone to failure. Prior to failure a GPU can produce indeterminate mathematical results, which are problematic and hard to spot. A focus on proper environmentals is very important for mathematical reliability as well as node reliability and throttling.

Integrating the most powerful GPUs, like the NVIDIA Tesla K80, provides little real value if data is unable to flow to and from the GPU and other system resources. For example, without sufficient bandwidth, considerable processing power goes underutilized while wait cycles accumulate. Insufficient system resource capabilities fail to keep the GPUs "fed" with data; compute node processes become stuck waiting for data to be returned. Much like any finely tuned machine, all parts need to equally access available resources at all times to run at peak efficiency. This differentiates vendors that design whole optimized systems, such as Cray, from those that assemble components.

Proprietary designs, while offering unique speeds and features to enable functionality "first to market," all too often cause their own set of obstacles. "Features" like specific drivers or unique hardware design requirements can hinder users' ability to keep their infrastructure at the forefront of IT evolution and stay competitive. In some cases, where standards are brushed aside in the pursuit of dominance in the moment, both near-term and future system component upgrades become difficult or impossible. Technology that was once proprietary and revolutionary becomes an obstacle that must be replaced or re-engineered, potentially at considerable expense.

Software must be carefully evaluated to ensure that the winning solution today is capable of running tomorrow's jobs. It is common to discover that a solution that worked well for a specific problem set does not accomplish other necessary tasks.

Cray CS-Storm as the GPU Platform of Choice

Although the challenges of operating a GPU-enabled cluster are different, the potential performance increases, cost savings, and ability to scale as workloads change and shift are hard points to counter.

Making the decision easier, the Cray CS-Storm high density, accelerator-optimized GPU-enabled clusters solve the unique challenges for meeting the demanding workloads facing the FS vertical in HPC, providing a stable, efficient, and powerful solution.

The CS-Storm cluster is the system of choice for the world's largest GPU-based supercomputers for its reliability, compute density, power efficiency, proper cooling, and balanced system performance at scale.

The Cray CS-Storm system provides fully integrated benefits ideal for FS operators. With a fully integrated hardware and software stack, CS-Storm systems save valuable administrator time, not requiring users to spend hours pre-installing core systems software or chasing down patches. BIOS integration with drivers, as for Infiniband, is handled at the factory. No need to find often-obscure patches to hardware bugs and integration issues for BIOS. Cray and NVIDIA's tight integration ensures that even under the most intensive 365/24/7 loads, GPUs are cooled properly so down-throttling (loss of performance) and mathematical indeterminism is avoided.

Since the CS-Storm system is what is called a "fat" GPU node, with eight GPUs, it can deliver huge performance benefits when the application is designed well to work with the amount of GPU processing power available. Careful application design is required to take full advantage of this. Cray not only creates the systems, but, as a supporter of OpenACC and OpenMP, is well placed to advise on application design.

With the Cray CS-Storm cluster supercomputers currently offering full integration of NVIDIA Tesla GPUs (K40/K80), they make perfect sense for FS industry experts looking to adopt a robust combination of traditional HPC hardware and advanced GPU integration. The performance enhancements, time-to-value, excellent TCO returns on datacenter utilization, infrastructure reliability, and overall maintenance costs from day one are a winning combination.

Appendix A – Cray CS-Storm Product Overview

As part of the Cray® CS™ cluster supercomputer series, Cray offers the CS-Storm cluster, an accelerator-optimized system that consists of multiple high-density multi-GPU server nodes, designed for massively parallel computing workloads. The system is available with a comprehensive HPC software stack including tools that are customizable to work with most open-source and commercial compilers, schedulers and libraries. Important facts to note about Cray CS-Storm include:

- Cray CS-Storm cluster rack can hold up to 22 2U rack mount CS-Storm server nodes
- Up to eight NVIDIA Tesla K40 or K80 GPU accelerators per node (PCIe Gen3 cards, up to 300W each) two host processors, delivering up to 1/3 GPU petaflops of compute performance in one 48U rack.
- With K40 accelerators, double-precision performance of up to 11.4 teraflops per blade
- With K80 accelerators, double-precision performance of up to 15 teraflops per blade
- Completely air-cooled platform without compromising system performance
- Available with liquid-cooled rear-door heat exchangers to maximize server density with minimal thermal impact to room
- Multiple interconnect topology options, including 3D Torus/fat tree, single/dual rail, QDR/FDR IB
- Cray Programming Environment on Cluster Systems (Cray PE on CS)
- Cray Tiered Adaptive Storage (TAS) and Cray® Sonexion® storage solutions available
- Customizable HPC cluster software stack options including multiple Linux® OS distributions, message passing (MPI) libraries, compilers, debuggers and performance tools