Gaining the Technology Edge

Five innovations every quant should know …

In the last decade technological advances have facilitated major innovations in the financial industry. Algorithms running on super-fast computers have enabled significant innovations and thus have revolutionized how algorithmic trading and high frequency trading is done. Simulation and back testing in risk management are vastly scalable due to innovations in parallel and distributed computing. Quants have been in the forefront of leveraging technology to enhance the performance of their models whether it is for speed, performance, or scalability. Every aspiring and practicing quant needs to be aware how technology can enhance their model development and deployment efforts to maximize the performance of their models. In the last decade, I have worked with more than 25 customers helping them optimize their code to develop robust, high-performance quantitative models. Technology can be a significant enabler and the benefits it can generate can be huge when optimally used. While optimizing code is important, code that is optimized to leverage the technology available reaps far more benefits. Time and again, I have realized that knowledge of the technologies and optimal deployment of technology within an organization helps to maximize the return on investment and to vastly improve performance in deployed models. Using the appropriate technology for the right use-case is important and many a times, quants need to redesign their prototype implementations to fully leverage the technology they intend to use. In this article, I will discuss five technological innovations in the last two decades that have revolutionized quantitative model development and deployment. Knowledge of these technologies and optimal deployment can help quants develop efficient models and reap significant benefits and competitive advantage that was just not possible a decade ago.

The Five Innovations

Twenty years ago, the internet was in its infancy, computers were excruciatingly slow, and memory and processing speed in computers were scarce resources. The cheapest available computer today at a local electronic store packs more power than some of the fastest available computers available to high-end researchers twenty years ago. The cost of computing has also dropped significantly enabling more rapid experimentation, faster design-to-deployment cycles and a race to innovate where technology is a key enabler. Gone are the days where quants had to run their models and take long coffee breaks or sometimes run models overnight to complete their

experiments. The alternative was to make significant assumptions, run fewer experiments, and extrapolate results which severely compromised the integrity of the models. These short cuts resulted in many failed experiments severely limited quants from exploring and trying out new innovations.

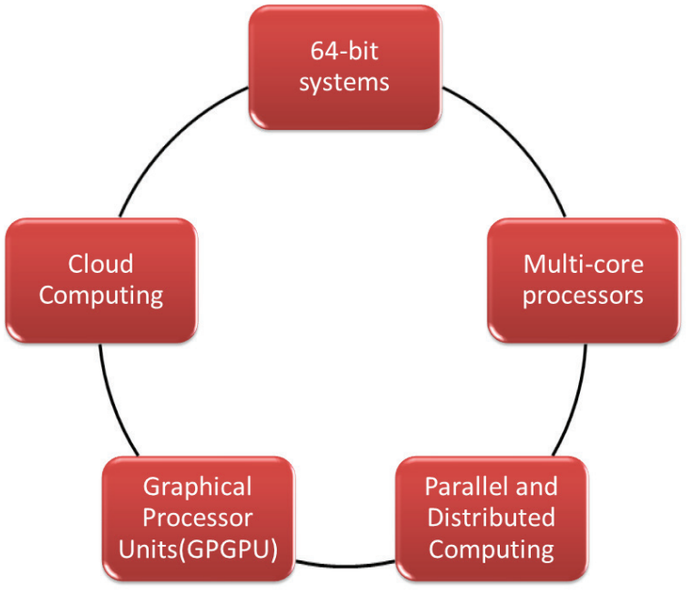
Fast-forward 20 years, computational hardware is cheap and innovations in hardware and software packages have lowered the bar for technological access. My smart phone has a Quad-core processor, 1 GB of RAM, and 32 GB of memory, which is much better in specifications than my expensive home computer I bought 15 years ago. Super-fast computers, cheap memory, faster networks and innovative software packages have made it possible to design and develop sophisticated quantitative models enabling quants to experiment more and focus on building more robust models. In the last decade, I have worked with more than 25 customers and helped them optimize their quantitative models. In most cases, clients approached us with scalability issues, performance overheads and at times, running the code was just not practical. In more than 60 percent of the use cases, my solution was to redesign the application to leverage technology to make applications run faster, scale better or to handle use cases that was not feasible without technology. Rather than complete redesign, testing and model revalidation efforts of just the codebase, leveraging technology with minimal code changes was an optimal path. In many cases, I evaluated the appropriate technology for the use case and illustrated how the technology can be used in the context of my client's application to optimize the performance of their models. In the following sections, I will discuss the five technologies (Figure 1) that I have used time and again for my customers. I will introduce and discuss the benefits of each technology and provide pointers and considerations using these technologies for their model development.

Every aspiring and practicing quant needs to be aware how technology can enhance their model development and deployment efforts to maximize the performance of their models

64-bit systems

In a recent conference, when I polled my audience (Windows users) on how many of them still used 32-bit operating systems, around 40 percent raised their hands. Most Windows Excel users are familiar with the 65,000 row limit or 2GB memory limit [4] when running on a 32-bit operating system. Many of my clients who run MATLAB on Windows 32-bit systems have seen "Out-of-Memory" errors when dealing with large datasets. The reason here is when running on a 32-bit operating system, the theoretical addressable memory available to the application running your code is 2^{32} bytes (4 GB) out of which applications like Microsoft Excel or MATLAB can access 2GB limiting the size of datasets that can be used in a quant model. By switching to a 64-bit operating system, the addressable space goes up to 264 bytes (up to 8TB) [5]. In the past, many of my clients running on 32-bit systems had to redesign their code to process data in chunks to address this scalability problem. By upgrading to 64-bit operating systems and by adding memory to computers which is very cheap, quants can benefit significantly by allowing processing of large datasets. However, at times, even if the operating system is 64-bit, the application drivers may still be designed for 32-bit machines. These may lead to a restriction of how large addressable memory which in turn means how large the datasets can be (despite having significant RAM on the machine). Even worse, driver compatibility issues (32 bit drivers on a 64 bit machine) can result in application errors. If you use tools that are built for 32-bit machines, check with your vendor/IT department on 64-bit support before upgrading your machines.

Figure 1: The Five Innovations



Multi-core processors

Moore's law [1] states that the number of transistors on integrated circuits doubles approximately every two years. In the last two decades, we have seen Moore's law realized in the processors developed by Intel and AMD with significant increases in processor speed and with multi-core

processors. A faster processor means that quants can run their code faster. Many times my clients see significant performance improvements by just upgrading to a faster computer. In addition, quants can leverage multiple processors on their machines to increase scalability. This can be accomplished by writing multi-threaded programs or by parallelizing their code (I will discuss this in more detail later). Writing multi-threaded programs or parallel programs from scratch isn't trivial. Quants must consider whether their program design can truly leverage multi-core capabilities. They must also consider whether the implementation implicitly leverages multi-core processors or should be explicitly implemented to leverage the multiple cores on a machine. In addition, developers must consider what else would be running on these machines and the resources available to their programs. Quants typically use C++ libraries to write multi-threaded programs. Boost is an example of one such library which supports multi-threading [2]. These libraries offer granular control and lets developers tune their applications. But developing multi-threaded programs is inherently complex and programs must be designed and tested thoroughly to avoid problems like race conditions. Higher-level languages like MATLAB offer implicit multi-threaded support for many functions [3] can also be used to leverage multi-core processors. The cautionary note here is programs that can leverage multiple cores on a machine may significantly enhance performance and scaling but is complex to design and implement.

Parallel and Distributed Computing

Quants have use cases where the same/similar operation needs to be repeated multiple times (for example, Monte-Carlo simulations or stress testing). Risk management and option pricing applications involving Monte-Carlo simulations can be classic use cases for parallel and distributed computing where each simulation can be run independently. Parallel computing enables each task (or simulation) to be run independently on different cores of a multi-core processor. This enhances scalability and enables the application to run faster. Distributed computing further enables leveraging clusters of machines or grids to run applications significantly faster thus increasing scalability. Depending on the type of problem, a combination of parallel and distributed computing methodologies can be used [6] to scale the problem implementation. Many times, I have been able to significantly enhance performance in my client models by leveraging parallel computing and with minimal changes to the program design using MATLAB and associated parallel computing tools [7]. Since most computers have multi-core processors nowadays, it is worth leveraging the available resources when applications are designed to run large number of simulations. However, quants must evaluate whether their application can truly benefit from parallel and distributed computing methodologies. One of the primary considerations is to determine the portion of the application that can be parallelized. Since there is overhead involved in passing data and synchronization, adequate design and testing must be done to evaluate if the computational benefits supersede the data transfer performance hit. In addition, different languages support parallel and distributed computing through various APIs. Quants must be comfortable designing programs optimally using these APIs to ensure that the computational resources are used optimally. Further, quants must consider the redesign effort necessary to leverage parallel and distributed computing technologies. Though parallel and distributed programming approaches offer huge promise in enhancing scalability, a thorough understanding of the methodologies in needed to use these tools properly. If you are new to these areas, consult someone familiar with these technologies to ensure optimal design and implementation before significant investment of time, effort and resources to build a parallel and distributed computing system.

Quants have use cases where the same/similar operation needs to be repeated multiple times (for example, Monte-Carlo simulations or stress testing)

General-purpose computing on graphics processing units (GPGPUs)

The last five years have significant improvements in GPGPU technology [9] and usage in the computational finance. GPGPUs or simply GPUs are optimized for massively-data parallel operations and have found uses in Monte-Carlo simulations, option pricing etc. Typically, applications leveraging GPGPU technology require sophisticated and dedicated GPU hardware. NVIDIA and AMD are major vendors who supply specialized GPU cards and associated drivers and API. Designing and developing GPU programs are usually non-trivial and requires significant understanding of the hardware stack, the API and the algorithm design aspects. For example, NVIDIA provides a CUDA toolkit [8] to leverage their GPU hardware. Application design and optimizing code targeted to the hardware architecture in use is essential for well-designed GPU programs. For example, CPUs are designed for low-latency access to datasets leveraging the L2 cache. GPUs on the other hand have a smaller L2 cache but have a huge number of processors optimized for data-parallel tasks. Typically, the program design involves writing code that transfers data from a CPU context to a GPU context for parallel processing and retrieving the results back to the CPU context for post-processing. The design also involves leveraging the CPU for massively parallel tasks whereas the CPU is used for other computational tasks. MATLAB provides GPU programming support through the Parallel Computing Toolbox [7]. Properly optimized code leveraging GPUs can obtain massive scalability on a desktop-sized hardware platform that in the past was only feasible through massive compute grids. Considering the complexity in design of the code customized for a GPU, we recommend quants to work with GPU application developers to minimize the design

and development lifecycle of the application and to develop optimal GPU code.

Cloud Computing

Cloud Computing has rapidly gained popularity in the last five years. Though cloud computing is more prevalent outside of financial services in areas like as e-commerce, the popularity in the financial industry has been increasing. With financial services companies opting for leaner IT infrastructure, more and more vendors are offering cloud-based solutions and financial companies are experimenting with running their applications on the clouds. One of the key benefits of cloud computing is its elastic nature and on-demand availability of resources that enables companies to scale their problems without having to buying hardware and software resources in-house. In addition, experimentation is easier on the cloud allowing quants to try models that require significant hardware. After trying models on the cloud and validating the results, organizations can procure dedicated hardware or leverage private cloud services to run their models. Many vendors like Amazon [10] offer a variety of hardware and software choices, including hardware with GPU cards, enabling quants to try out the technologies described above. In addition, many financial companies are experimenting with Big Data applications for which cloud computing provides the scalable hardware and software resources to run large applications. Concerns such as data and application security, network overhead etc. are still open problems that have hindered rapid adoption of Cloud computing in the financial services industry. However, with vendors like Google, Microsoft, Amazon and Oracle making significant investments to address these issues as grow their offerings, cloud computing is destined to be the next wave of the future. Familiarity with the cloud computing model would help quants leverage these massively scalable technologies to develop and try innovative applications providing them with flexibility and access to hardware/software choices that were never available to them in the past.

Conclusion

Advances in technology in the last two decades have significantly enhanced the toolsets quants have to develop, test and scale innovative quantitative applications. Technology has been a significant enabler for quants to develop applications that scale and perform in ways that was never possible a decade or two ago. In this article, I have discussed five technological innovations, 64-bit systems, multi-core processors, parallel and distributed computing, GPGPUs and cloud computing that are being used by quants round the world to gain competitive advantage in the market place. I have used these technologies with various clients and my clients have seen significant benefits with these technologies. However, every opportunity comes with new challenges. Each technology I discussed must be thoroughly understood and used appropriately to gain maximum benefit. By optimally designing applications to leverage these technologies, quants can speed up and advance their innovations and bring their research into the market place.

Technology has been a significant enabler for quants to develop applications that scale and perform in ways that was never possible a decade or two ago

REFERENCES

http://en.wikipedia.org/wiki/Moore%27s_law
http://www.boost.org/doc/libs/1_38_0/doc/html/thread.html
http://www.mathworks.com/discovery/multicore-matlab.html
http://msdn.microsoft.com/en-us/library/office/ff700514%28v=office.14%29.aspx
http://www.mathworks.com/products/matlab/preparing-for-64-bit-windows.html#6
https://en.wikipedia.org/wiki/Distributed_computing
http://www.mathworks.com/products/parallel-computing/
http://www.nvidia.com/object/cuda_home_new.html
https://en.wikipedia.org/wiki/GPGPU
http://aws.amazon.com/ec2/

About the Author

Sri Krishnamurthy, CFA, CAP, is the founder of QuantUniversity.com, a data and quantitative analysis company. He has significant experience in designing quantitative finance applications for some of the world's largest asset management and financial companies. Sri has worked at MathWorks, where he worked with more than 25 customers providing asset management, energy analytics, risk management, and trading solutions. Prior to that, Sri was a consultant at Endeca (now Oracle) in their Analytics Group and at Citigroup in their Fixed-Income Group. He has a MS in Computer Science and an MS in Computer Systems Engineering, both from Northeastern University and an MBA from Babson College. Sri is also an Adjunct Lecturer at Babson College and teaches an in their MBA program. He is the author of the forthcoming book published by Wiley titled *Financial Application Development with MATLAB*. He can be reached at sri@quantuniversity.com .