Calculation of Classical Constants and Special Functions for Fun and Profit: Hardware View

D. V. Chudnovsky, G. V. Chudnovsky
IMAS
NYU Tandon School of Engineering
6 MetroTech Center
Brooklyn, NY 11201

May 14, 2019

It is About Hardware

We want to thank the organizers of this conference, Alin and Kilian, for their kind invitation.

Brief look at the problem of calculation of classical constants with very large precision, large scale simulations that evaluate values of special functions.

Strictly hardware point of view.

Life with a "Hardware Disease"

Some time in the mid 1980s we had been bitten with a "lethal" hardware bug.

In the last 20 years we have been mostly working on designing chips for contemporary supercomputers, built by IBM.

The current generation chips are manufactured in 14 nm process.

Cyclops

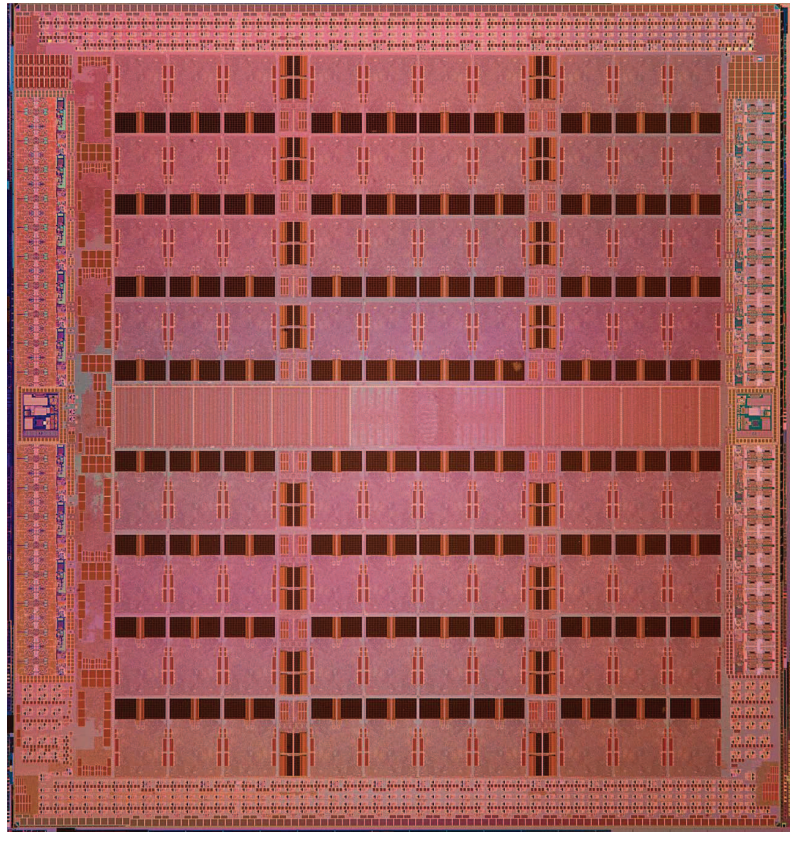

Figure: Cyclops Chip; 160 CPU Cores

Exponential Growth

Astonishing improvements in chips performance.
Almost exactly thirty years ago this day, owe have run computation of decimal digits of π on Cray-2 supercomputer with slightly under 2 Gigaflops performance.
Today the Apple Watch exceeds this performance.
Typical supercomputer operates at 10 to 100+ Petaflops $(10 \cdot 10^{15} - 100 \cdot 10^{15})$.
Computational objects in the cloud, are in tens of Teraflop (10^{12}) performance – Intel/AMD chips with GPU accelerators: NVIDIA, ATI, Intel, Xilinx.

Exponential Growth, So Far

Still have room for improvement before quantum computers or such.

Getting ready to scale to 3 nm features in the coming decade.

Potential of up to X16 density improvement (at least for memories).

Would be enough for our remaining lifetime.

But Is it Properly Working? The Pi Test.

What do you do with such a wonderful amount of compute?

Check that these systems with quadrillions (10^{15}) of transistors actually work.

Stress tests are needed.

Sexiest tests of the hardware involve the calculation of π with the "gazzilion" of digits.

Perhaps, many programmers and system managers are mathematicians.

This approach to hardware testing goes back to John von Neumann.

Someone has to do it

The ultimate users of systems that we have participated in designing, insisted that the tests involved π calculation either in the form of a giant string of decimal digits or in the BPP-style calculation of a window of binary digits of π.
This tests the programmability, endurance and the recovery of supercomputers.

New Record Calculation

In the public domain the race for the calculation of decimal/binary digits expansion, or for a window of binary digits of π is going unabated worldwide.

The most recent record of calculation of decimal/binary digits expansion of π belongs to Emma Haruka Iwao from Google (Pi day of 2019) who computed 31.4 Trillion decimal digits of π with a companion BPP window computed by Alexander Yee.

The computation of Emma Haruka Iwao had been carried out during 121 days on a Google cluster of 24 servers with 100 Xeon Skylake Platinum cores.

One Hypergeometric Identity

The algorithm for this calculation was based on a parallel implementation of our favorite π Ramanujan-like identity (1988)

$$\sum_{n=0}^{\infty} (-1)^n \cdot \frac{(6\,n)!}{(3\,n)! \cdot n!^3} \cdot \frac{163096908 + 6541681608 \cdot n}{(262537412640768000)^{n+1/2}} = \frac{1}{\pi}$$

arising from the 9th class one negative discriminant -163.

Nature of Errors

During this calculation single-bit memory errors were protected by the ECC (SEC+DED codes).

There were two uncorrectable errors.

Errors of Nature

A robust design guarantees that power supply, network and storage problem would be detected and repaired automatically in hardware. The ECC protects top parts of the memory hierarchy – DRAMs, L3, L2, L1 caches.

Then why we still expect some failures in a large machine?

The cause of the failure in chips is a radiation impact, generated by cosmic rays and alpha particles.

Still some very minor imperfections in wafers.

Soft Error Rate – FIT

3D FinFET transistors reduce the impact, and the soft error rate is now by an order of magnitude less than it was 10 years ago, but the density had increased X32.

FIT – Failures-In-Time per billion hours – is now of the order of 40 FITs per million states.

In addition, there are transient errors in combinational logic gates.

It Could Have Been Worse

We would have expected in the system of about 100 Xeon cores the number about 4 uncorrected errors during 100 days of non-stop calculation.
This "soft error rate" holds for the sea level, increasing by a factor of 2.3 per every km of altitude on Earth.
Do not compute π while flying to Mars.

Errors in Large Systems/Computations

Not everything can be ECC protected due to the area and timing requirements, and thus every very large supercomputer is expected to have a large number of fails in a long run.
Large chips (multi-core processors, router, etc.) have these days more than 100 million latches, and large systems can have tens of thousand of such chips.
This problem is well described in DARPA's Exaflop report from several years ago.
Calculations of π thus provide us with a wonderful opportunity to test large computers, particularly because in integer based algorithms, each step can be checked.

Checks

According to A. Yee, the author of the y-cruncher tool that is used in recent π calculations, a large number of errors past the mechanical fails, and errors detected by the ECC, was caught by a variety of modular checks.

Massive π calculations, despite their practical value, can be considered fan, because programmers, computer operators and the general public seem to enjoy it.

Hardly one can make a profession out of it.

What HPC was doing?

HPC is high performance computing=supercomputing.
Until recently, mostly DOE and Grand Challenges computations, primarily for physicists, biologists, and chemists.
Mathematicians were able to fit in large supercomputations, often providing the last punch to solve a long standing problem with a big run of calculation of special cases.

What HPC is doing?

Few years ago the amount of the bit/crypto coin mining became the one of dominants in HPC as well as in individual platforms.
Also grows in molecular biology computations and financial applications.
However, the most recent heavy users in HPC are in: ML (machine learning) and AI.

"Arms Race" in Calculations

Where in this picture the calculations with standard mathematical objects fit? If you look at hypergeometric functions and related objects from number theory and algebraic geometry, the best place to find them is in the mathematical finance.

"Arms race" in finance calculations, that started about a decade ago.

Unlike the case of large scale calculations (like π), here the latency is all important.

Here the race goes to the swift.

This creates a unique environment, where the quantity and quality of computations are married to the latency.

HFT

HFT is high frequency trading, where the round-trip latency is a few μs (10^{-6} sec).
Most of the latency goes to the communication to and from the "trading floor", and the rest (on the order of 750 ns) is used for the decision of bid/ask/buy/sell/hold.
Communication part of the latency here is measured by the speed light and the distance to the server on the trading floor.
Special purpose hardware, mostly based on FPGAs, is here, for multi-channel 10G communication support, and ML implementation of the "decision logic".
Not much time is left to do heavy duty mathematics.

Mathematical Finance

Nice mathematics is in longer computations for portfolio and risk management.
Required by law after 2008 "computational debacle". Interesting subject of complexity, stability and coverage of the computations.
Let us look at the lowest level needed in all these analytics. It inevitably involves massive compute with various probability distributions trying to estimate the short term and long term events impact.

Monte Carlo Trees and Forests

Classical Black-Scholes style of computations uses Monte Carlo in the form or random trees, and forest runs.
Various versions of the Monte Carlo and quantile analysis are used for such tasks as risk management and handling of various complex financial instruments (options, bonds, futures,...).

Classic Probabilistic Distributions

Often Monte Carlo and quantile evaluations are focused on the parametric probabilistic distributions, where we meet some special functions:
Normal, Cauchy, exponential, Laplace, Pareto, Weibull, Frechet, Beta, Gamma, χ^2, Student T.
These distributions and their multivariate versions are often married to empirical distributions.

High Quality Digital Samples

Need high quality random samples and quantile analysis data for these distributions.

John von Neumann thought about it in a memo to Stan Ulam on May 21, 1947, as a reply to suggestion by Ulam to use the Monte Carlo sampling.

He envisioned two methods of generating samples:

an inversion of CDF, or a rejection method.

In both cases one has to start with the uniform pseudo random number generator.

Speed up Calculations

Both methods relay on many evaluations of special functions, at least for the parametric distributions. This puts a heavy computational load and significantly increases the latency of the analysis.
In the last 30 years an approach to the inverse method has been based on the approximation of the inverse function in its different regions.
We know that computation of solutions of l.o.d.e.s with rational function coefficients, based on recurrences of their Taylor expansions and Pade approximations are most efficient for the calculation with large (unlimited) precision for example, in the hypergeometric identities for π.

Cost of Approximations and Calculations

However, the price of large scale computations often changes the opinion of the usefulness of the distribution. E.g., the stable distributions – with the characteristic function of

$$e^{it\mu - |ct|^{\alpha}(1 - i\beta sgn(t) \tan \frac{\pi\alpha}{2})}$$

(for $\alpha \neq 0$) related to incomplete hypergeometric function, advocated by B. Mandelbroit for the use in financial markets, were replaced by the Student T distribution with the CDF of

$$(1 + sgn(s)(1 - I_{\frac{\nu}{x^2+\nu}}(\nu/2, 1/2))/2$$

because it was easier to compute.

Floating Point Calculations

IEEE floating point double precision calculation use minmax rational approximations in different regions of the inverse function.
The commercial hardware used for that ranges from Intel/AMD server platforms to GPU cards NVIDIA/ATI and, recently, Xilinx FPGA.
Compute engines are floating point intensive cores that evaluate polynomial/rational functions approximations.
The division is implemented by the iterative methods to save the silicon area.
Here the floating point additions can be very fast up to 5 GHz, multiplication not so.

Hardware View

In the current 14 nm technology, the area of the double precision floating point multiply accumulate unit, together the accumulation registers, takes less than
$0.016 mm^2$ in area,
when operated at 1 GHz.
The largest chip (reticle) is 26 x 31 mm.
Thus, if one can afford to build such chips, one can have tens of thousands of double precision floating point engines together with embedded memories to support them.
If one builds such chips very well, one gets oneself a NVIDIA-style accelerator.

The Bottleneck

Main flaw - these engines cannot operate independently for the lack of resources to communicate with each other and the outside world.
The reason for this bottleneck is:
while the area decreases quadratically with the feature size, wires can decrease at most linearly.
For example, in 14 nm technology the minimal pitch between the tiniest of wires is 64 nm.
As a result most GPU and similar massive accelerators have to operate in SIMD – single instruction multiple data – mode.
This creates performance problems for Monte Carlo calculations.
We recommend a series of paper of W. Shaw et al. on "the Quantile Mechanics", dealing with GPU programmability.

Approximations

From the point of view of the approximation theory we are still on the steady path started in the late 1950s, with better computer implementation of higher order approximations.
We would like the old fashion processor-in–memory approach, where we can load and store from/to large memories the data on the same or the next cycle.
Looking for good compromises.
Till there was no commercially viable high-bandwidth solution for integrating large memories with large computational chips.
E.g. the Memory Cube from Micron failed in the marketplace.

New Memory Solution

A possible long-term winner: HBM/HBM2E – High Bandwidth Memory – from Samsung and Xynix.
Each module has 3D stacked memories dies of 8 or 16 GB, and 300 GB/sec bandwidth with 8 separate channels.
Serious difficulties of integration with 3500 pads per module, but a great solution for the rapid access to huge external data.
Large chips can have up to 4 tightly integrated HBM modules.
This opens an interesting possibility of having large table look-ups for massive calculations, very useful for the quantile analysis of empirical distributions .

Aquabolt HBM2

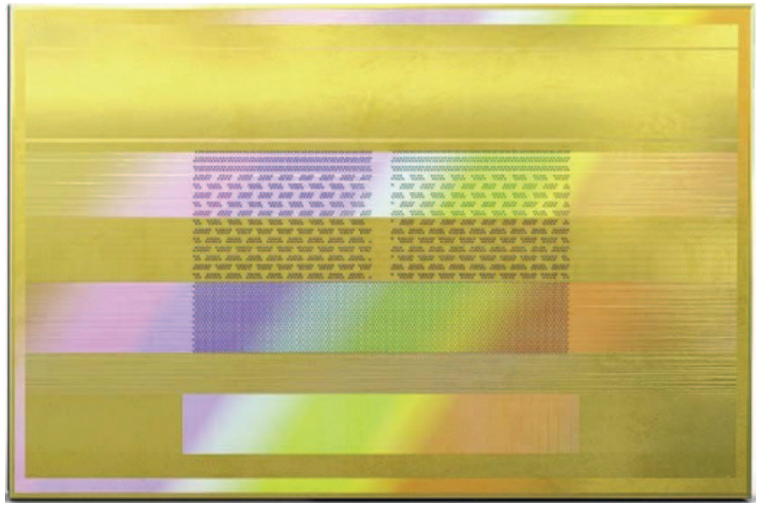

Figure: Samsung Aquabolt HBM2; 3500 microbumps

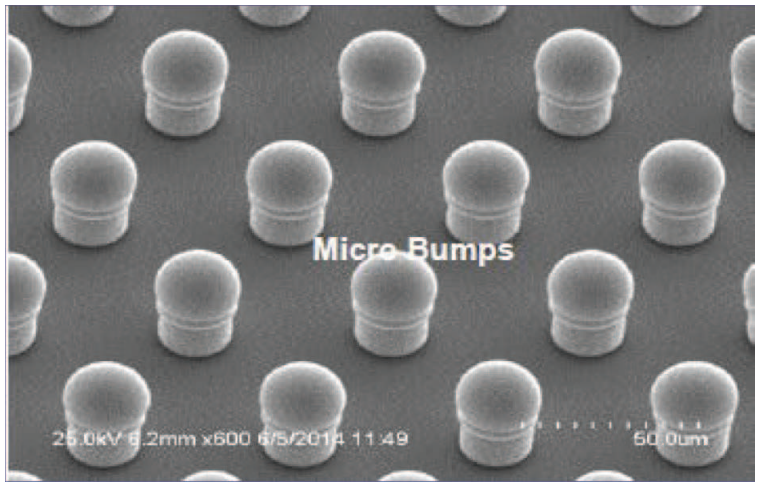

Figure: Microbumps with 48 X 27.5 μm^2 pitch

The Scope of Calculations

Large scope of calculations for the risk assessment and option price projection, for example.

The regulatory requirements need the calculation time to be at most on the order of hours for 100,000s to 1,000,000 positions.

These calculations use clusters with equivalents of tens of thousands of Xeon cores and extra NVIDIA accelerators.

One cannot run calculations dismissing the long tales of the distributions (which would greatly accelerate the approximation computations).

Rare, but "Deadly" Events

One weird reason for the 2008 crush was a claim that it was due to Monte Carlo programs that chopped off the long tails of the distributions, because 6+ σ- events could have been ignored.
Events even with a higher σ often happen in financial markets.
In our chip design the requirements are significantly stronger, to deal with glitches in voltage.
For example, every single embedded memory is designed to be protected against 7σ event.

Future Requirements

What the ultimate consumers of the risk management runs want to see – is the latency of under a minute on massive calculations involving Monte Carlo analysis, but without any cheating.

Additional interesting mathematical questions here involve the low discrepancy sampling to improve the complexity and accuracy of the Monte Carlo style computations.

Perspectives

The work in this area is rewarding, at least in one obvious respect. Ignoring the social implications, this field is also blooming with new research opportunities, when one combines Monte Carlo analysis with ML tools to shorten the prediction time, and, maybe, protect against the likely future financial and engineering glitches.
We personally are still looking forward to building new hardware, blissfully ignoring its uses, but hoping that ultimately mathematicians will benefit from them, one way or another.

Some References

D.V. Chudnovsky, G.V. Chudnovsky, Approximations and Complex Multiplication According to Ramanujan, in "Ramanujan Revisited", AP, 1988.

D. Harvey, J. Van Der Hoeven, Integer multiplication in time $O(n \log n)$, Mar. 2019.

P. Afshani, C. Frekseny, L. Kammay, K. Green Larse, Lower Bounds for Multiplication via Network Coding, Feb. 2019.

G. Steinbrecher, W. Shaw, Quantile Mechanics 1, 2007; N. Brickman, W. Shaw, Quantile Mechanics 2, 2010; A. Munir, W. Shaw, Quantile Mechanics 3, 2012.

S. Arora, B. Barak, M. Brunnermeier, R. Ge, Computational Complexity and Information Asymmetry in Financial Products, 2012.